Introduction to Convolutional Networks

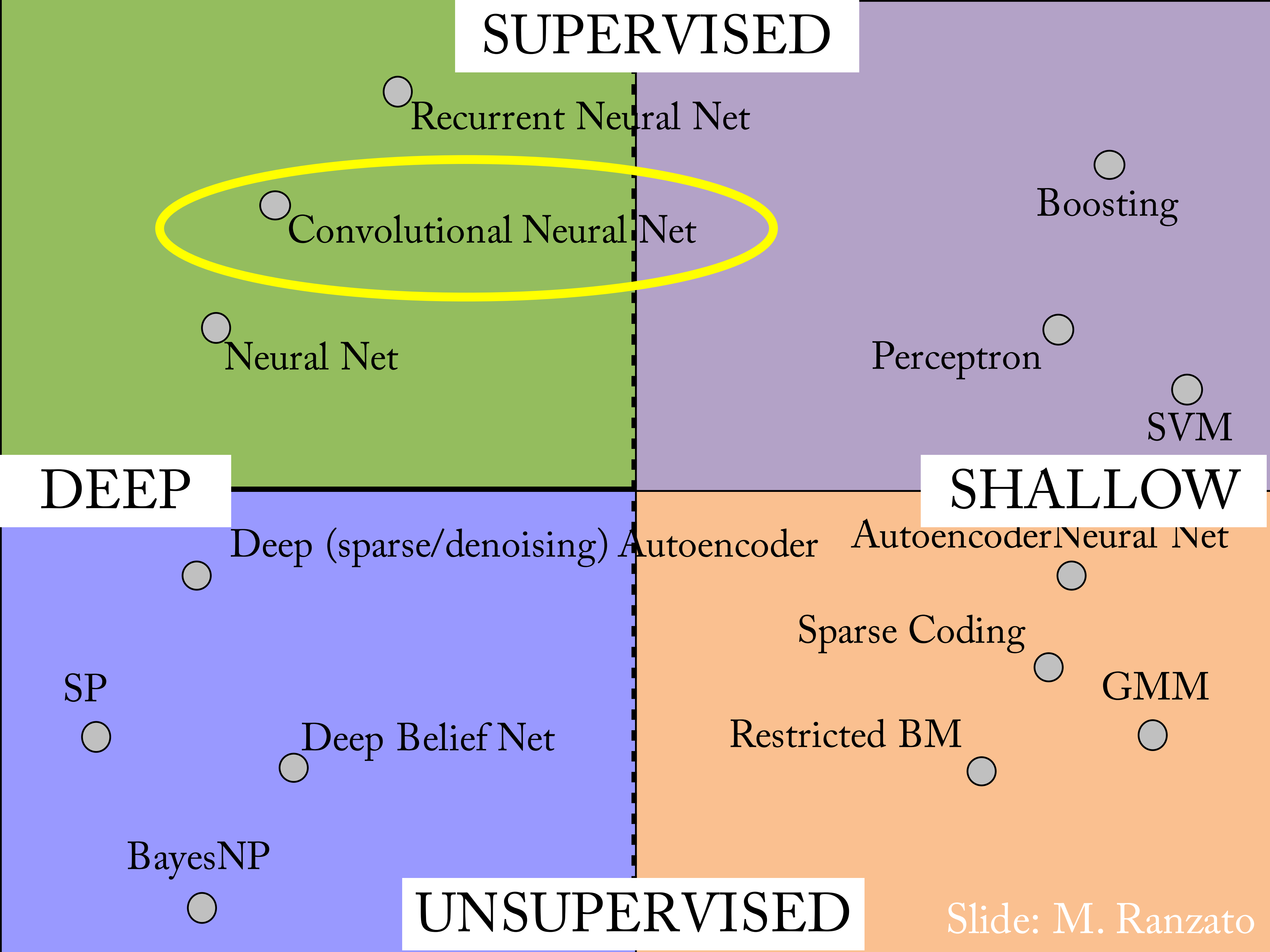
CIFAR Summer School 2016

Rob Fergus

Facebook AI Research
New York University

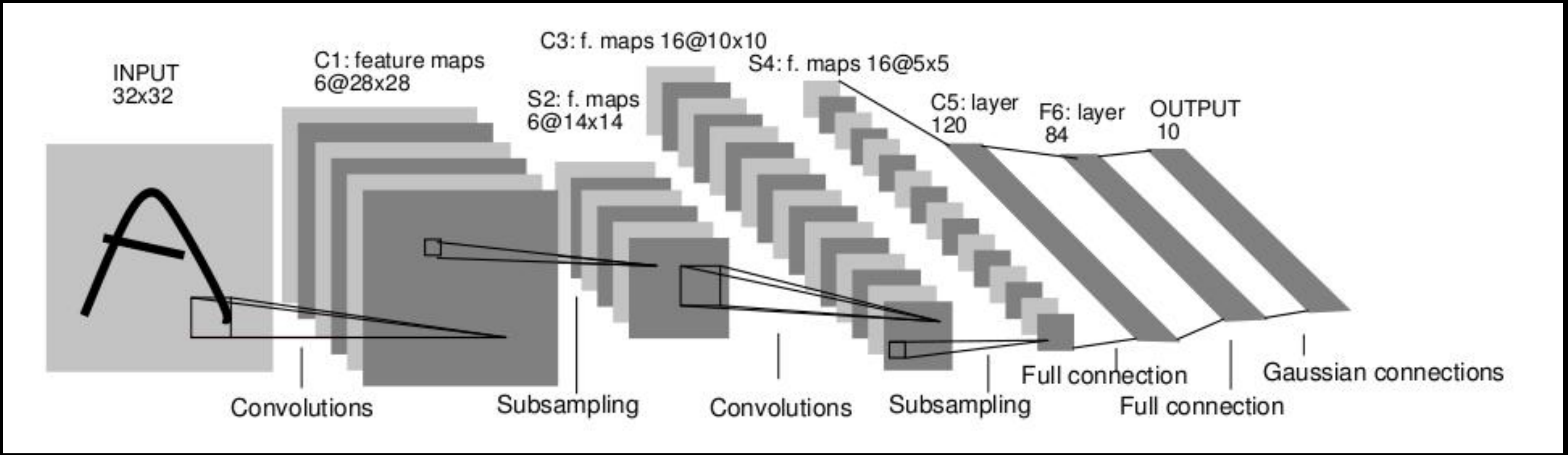
Overview

- Look at some of the recent progress with Convolutional Network models
  - Assume familiarity with basic neural nets
- Non-exhaustive coverage
  - Huge number of recent papers
- Review some computer vision applications


SUPERVISED
Recurrent Neural Net
Boosting
Convolutional Neural Net
Neural Net
Perceptron
SVM
DEEP
SHALLOW
Deep (sparse/denoising) Autoencoder
Autoencoder Neural Net
Sparse Coding
SP
GMM
Deep Belief Net
Restricted BM
BayesNP
UNSUPERVISED
Slide: M. Ranzato

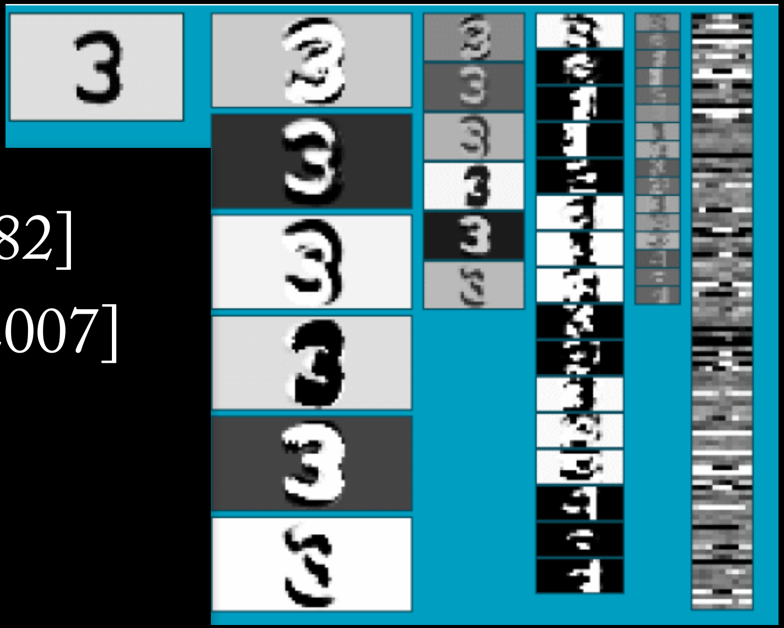
Convolutional Neural Networks

- LeCun et al. 1989
- Neural network with specialized connectivity structure



Multistage Hubel-Wiesel Architecture

- Stack multiple stages of simple cells / complex cells layers
- Higher stages compute more global, more invariant features
- Classification layer on top



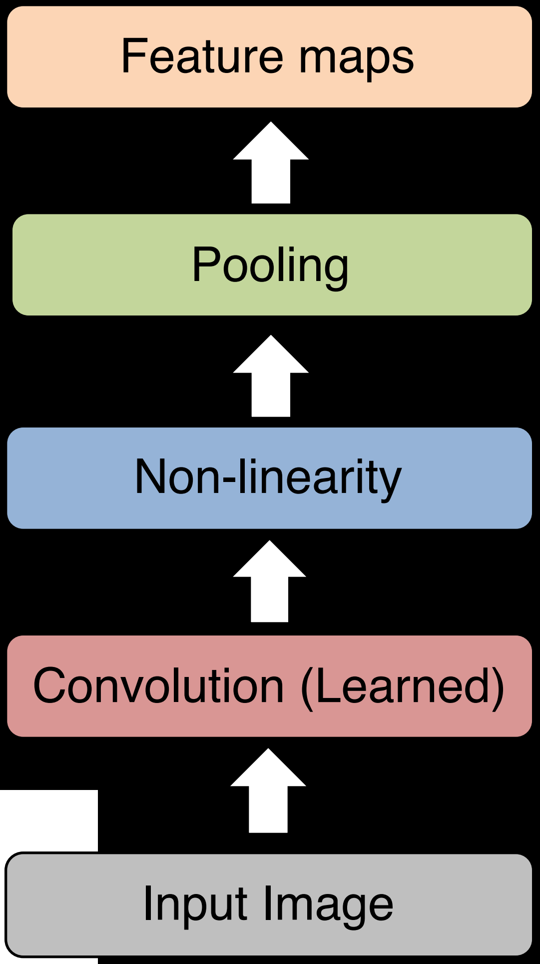
History:

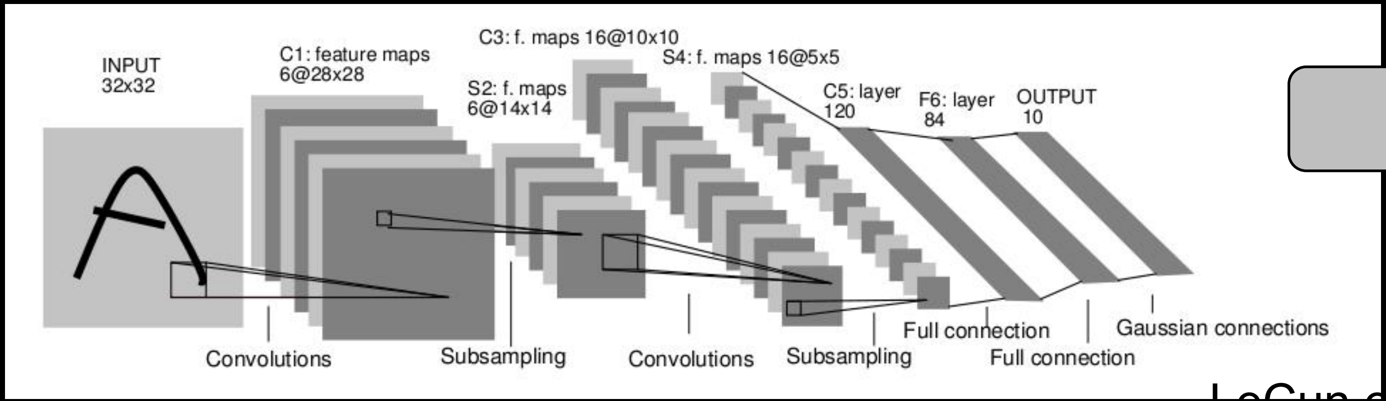
- Neocognitron [Fukushima 1971-1982]
- Convolutional Nets [LeCun 1988-2007]
- HMAX [Poggio 2002-2006]
- Many others….

Slide: Y.LeCun

Overview of Convnets

- Feed-forward:
  - Convolve input
  - Non-linearity (rectified linear)
  - Pooling (local max)
- Supervised
- Train convolutional filters by back-propagating classification error





Convnet Successes

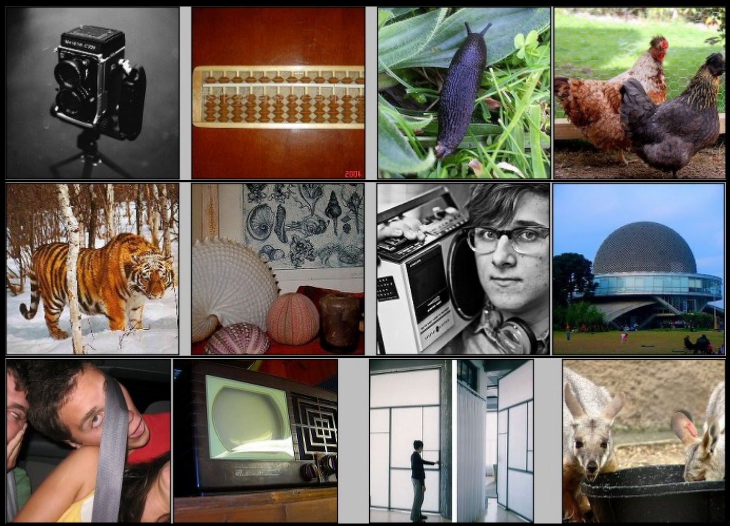
- Handwritten text/digits
  - MNIST (0.17% error [Ciresan et al. 2011])
  - Arabic & Chinese [Ciresan et al. 2012]

- Simpler recognition benchmarks
  - CIFAR-10 (9.3% error [Wan et al. 2013])
  - Traffic sign recognition
    - 0.56% error vs 1.16% for humans [Ciresan et al. 2011]

- But less good at more complex datasets
  - E.g. Caltech-101/256 (few training examples)

Application to ImageNet

IMAGENET



[Deng et al. CVPR 2009]

- ~14 million labeled images, 20k classes
- Images gathered from Internet
- Human labels via Amazon Turk

ImageNet Classification with Deep Convolutional Neural Networks [NIPS 2012]

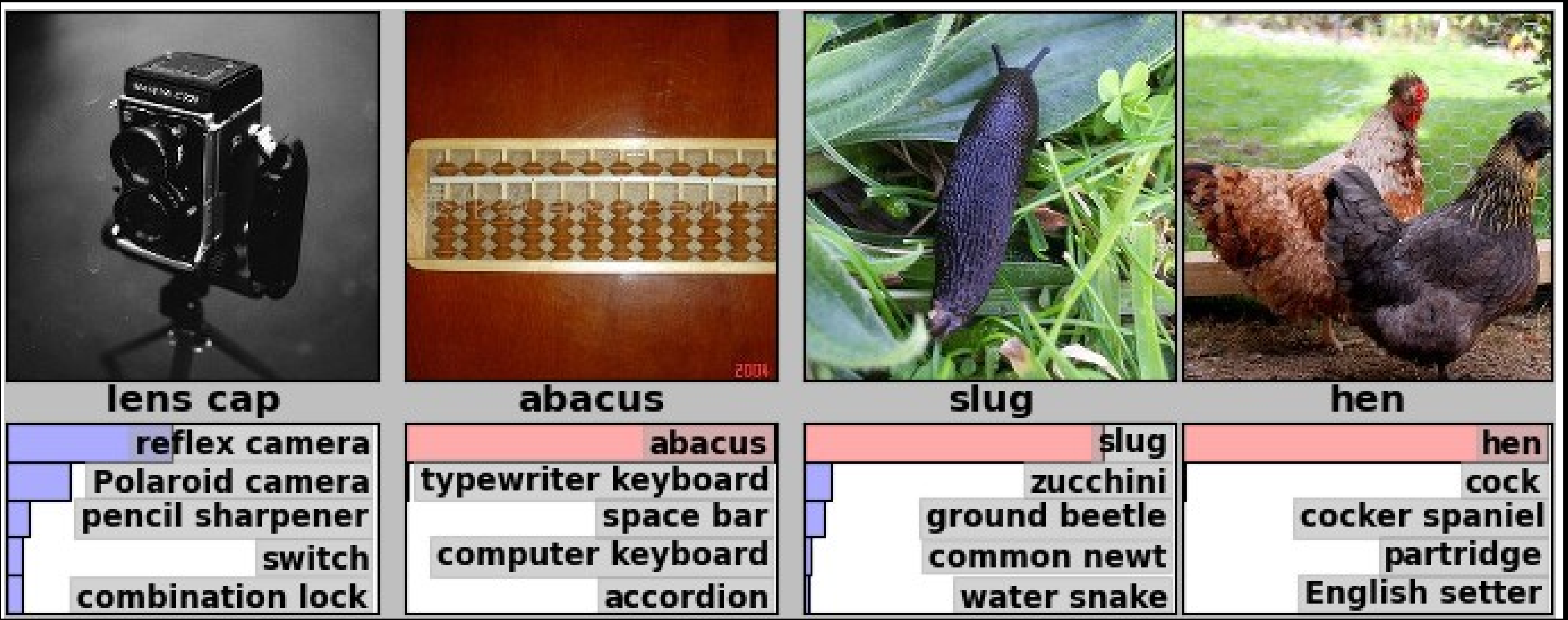
Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

Goal

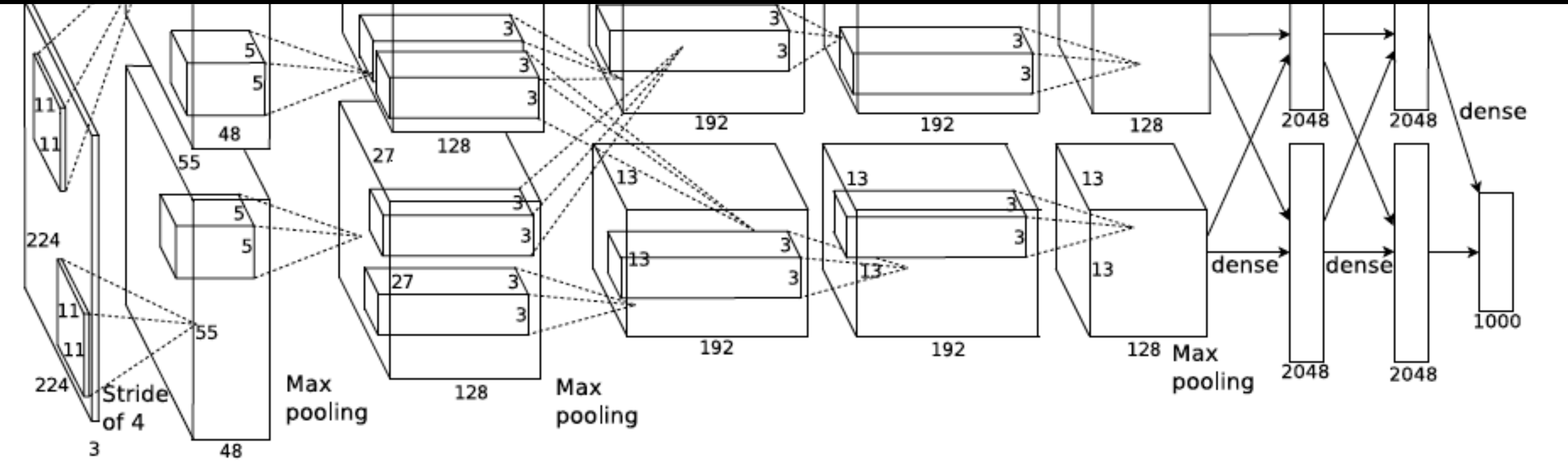
- Image Recognition
  - Pixels → Class Label



[Krizhevsky et al. NIPS 2012]

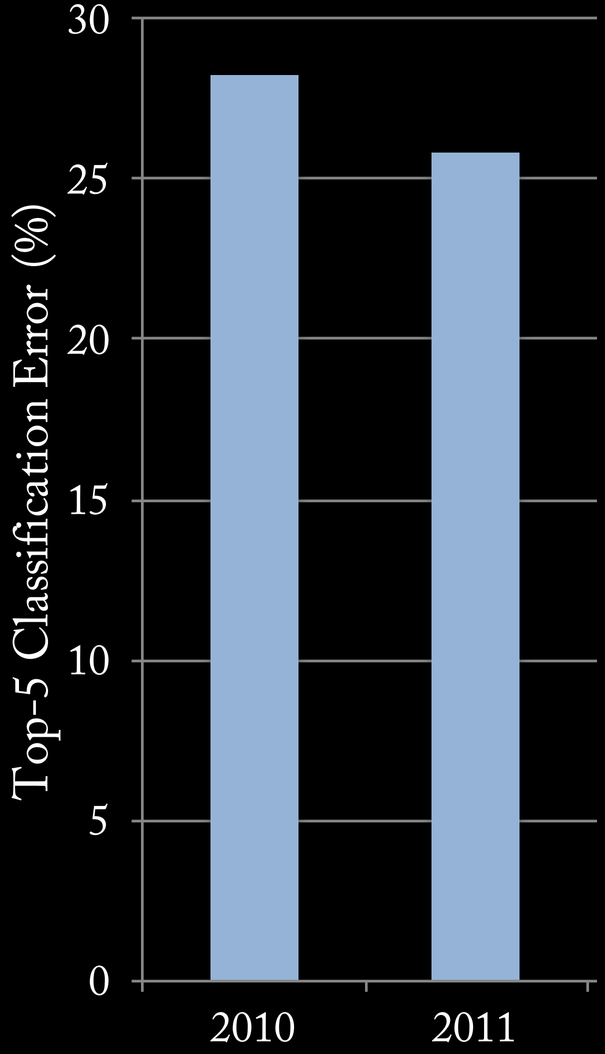
Krizhevsky et al. [NIPS2012]

- Same model as LeCun'98 but:
  - Bigger model (8 layers)
  - More data (10^6 vs 10^3 images)
  - GPU implementation (50x speedup over CPU)
  - Better regularization (DropOut)



- 7 hidden layers, 650,000 neurons, 60,000,000 parameters
- Trained on 2 GPUs for a week

ImageNet Classification (2010 – 2015)



Examples

- From Clarifai.com

Predicted Tags:

food	(16.00%)
dinner	(3.10%)
bbq	(2.90%)
market	(2.50%)
meal	(1.40%)
turkey	(1.40%)
grill	(1.30%)
pizza	(1.30%)
eat	(1.10%)
holiday	(1.00%)

Stats:

Size: 247.24 KB

Time: 110 ms

Examples

- From Clarifai.com

Predicted Tags:

ship	(2.30%)
helsinki	(1.80%)
fish	(1.40%)
port	(1.10%)
istanbul	(1.10%)
beach	(1.00%)
denmark	(1.00%)
copenhagen	(0.90%)
sea	(0.80%)
boat	(0.80%)

Examples

- From Clarifai.com

Predicted Tags:

barcelona	(6.50%)
street	(3.00%)
cave	(2.20%)
sagrada	(1.90%)
old	(1.80%)
night	(1.40%)
familia	(1.40%)
jerusalem	(1.40%)
guanajuato	(1.10%)
alley	(1.00%)

Stats:

Size: 278.96 KB

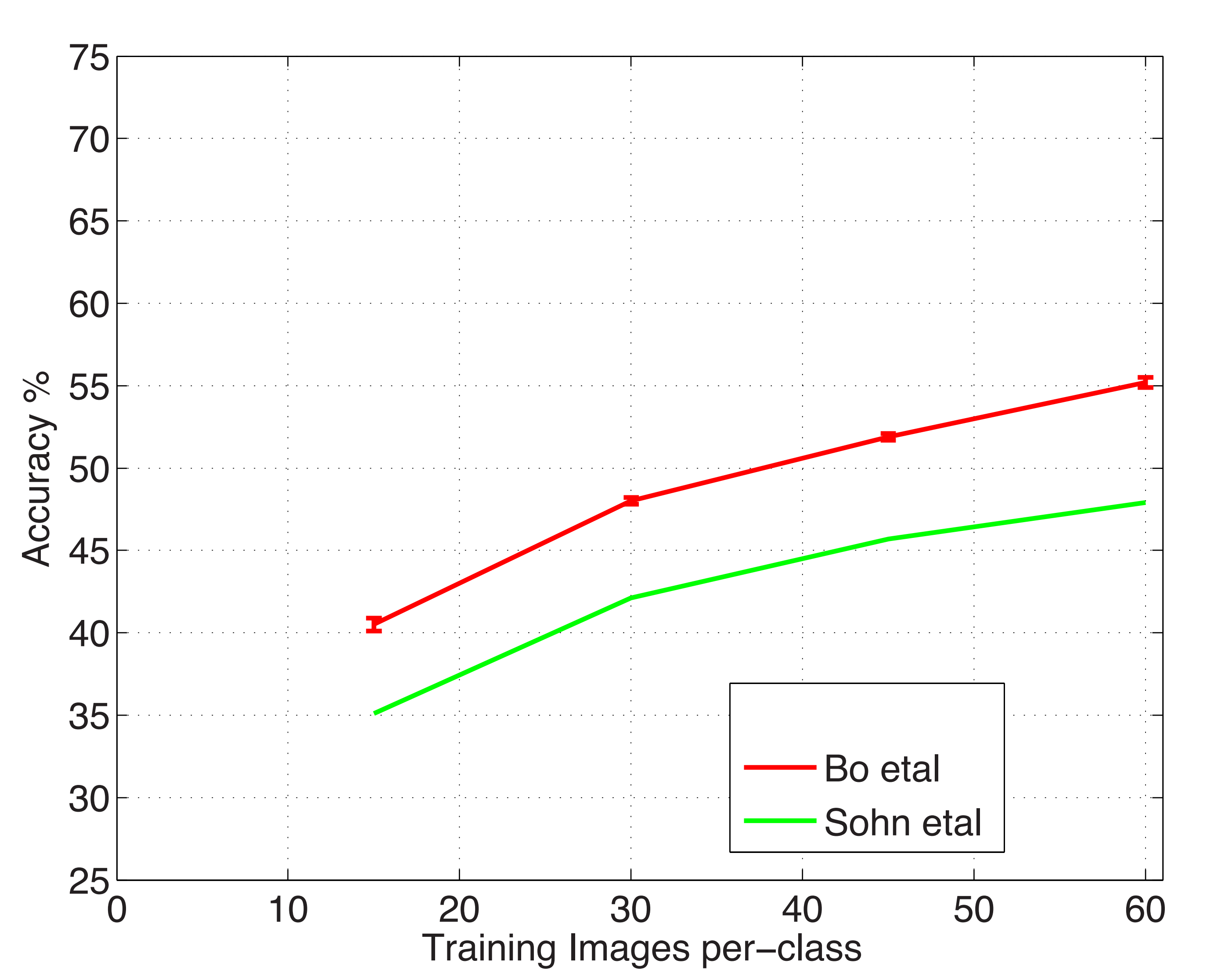
Time: 113 ms

Using Features on Other Datasets

- Train model on ImageNet 2012 training set
- Re-train classifier on new dataset
  - Just the top layer (softmax)
- Classify test set of new dataset

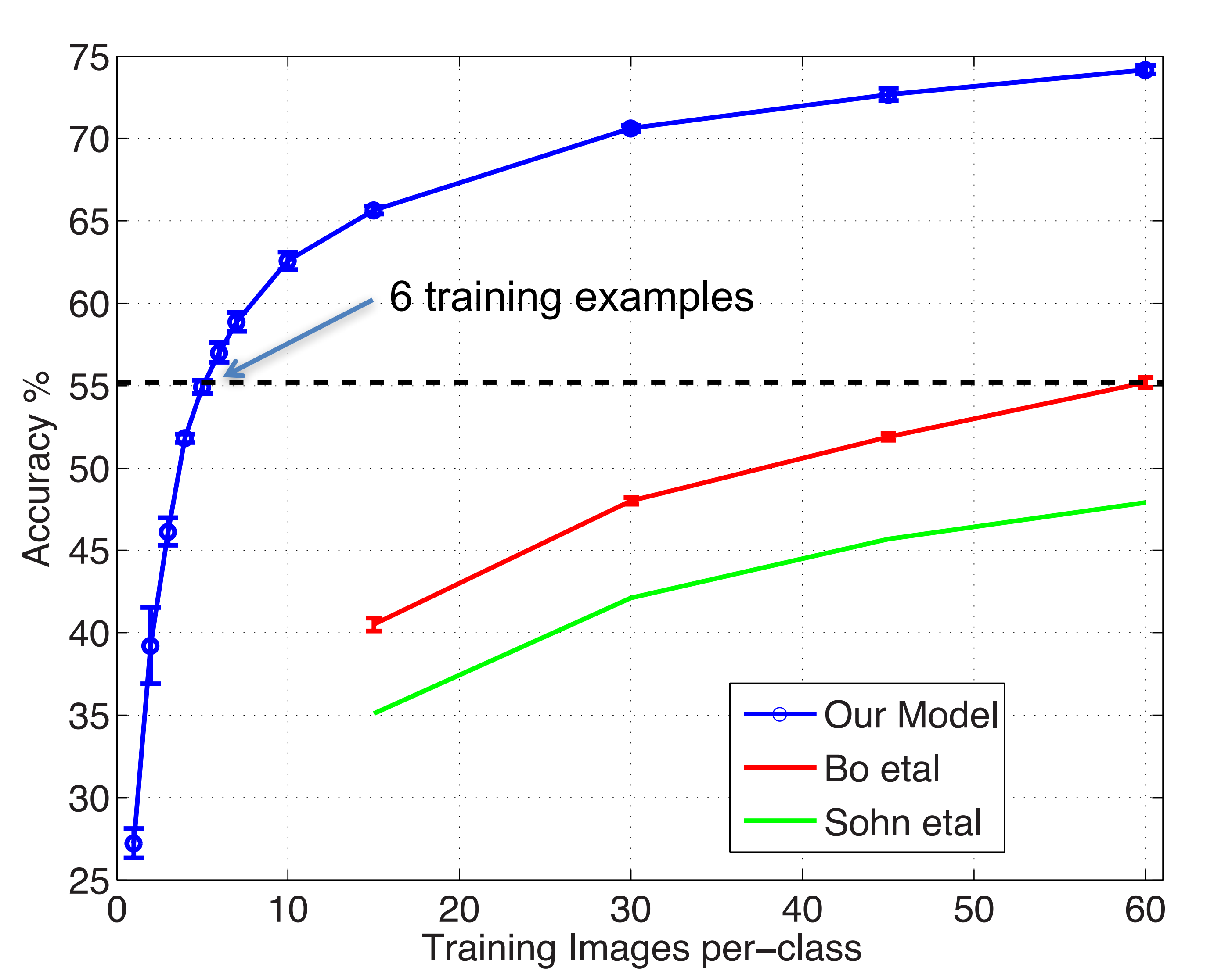
Caltech 256

Zeiler & Fergus, *Visualizing and Understanding Convolutional Networks*, arXiv 1311.2901, 2013



Caltech 256

Zeiler & Fergus, *Visualizing and Understanding Convolutional Networks*, arXiv 1311.2901, 2013



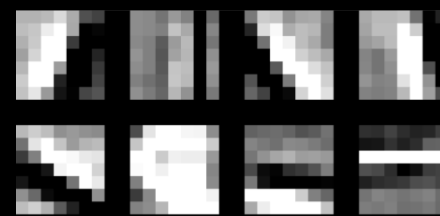
The Details

- Operations in each layer
- Architecture
- Training
- Results

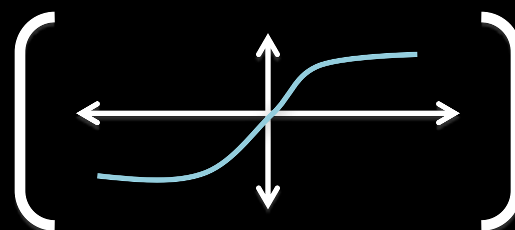
Components of Each Layer

Pixels / Features

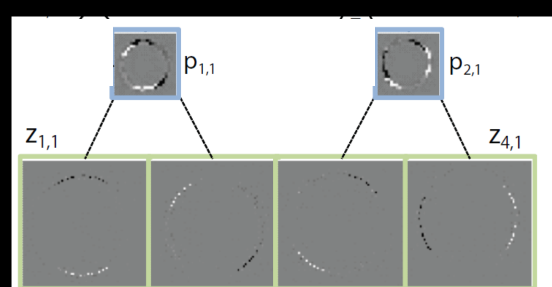
Filter with learned dictionary



Non-linearity



Spatial local max pooling



Output Features

Filtering

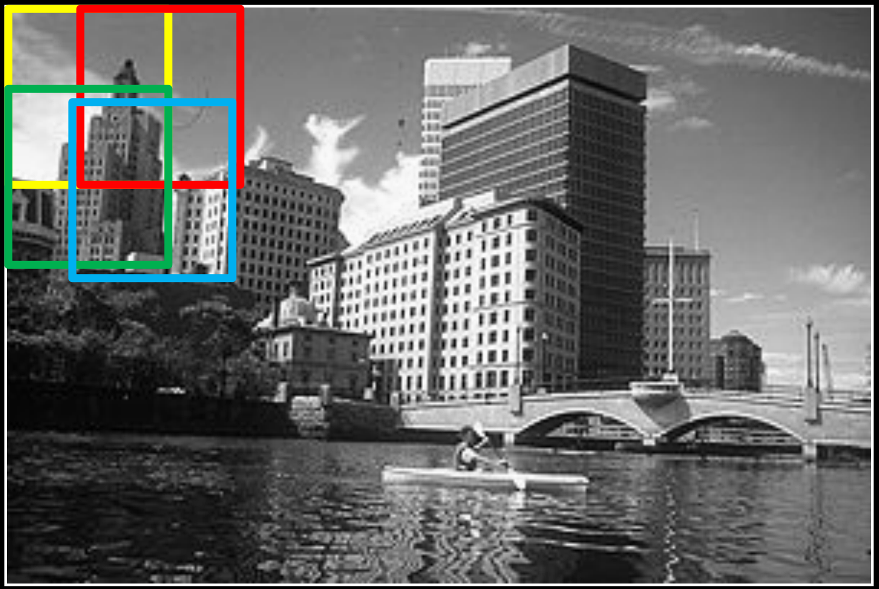
- Convolution
  - Filter is learned during training
  - Same filter at each location

Input

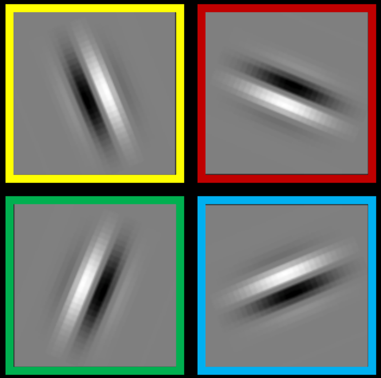
Feature Map

Filtering

- Local
  - Each unit layer above look at local window
  - But no weight tying

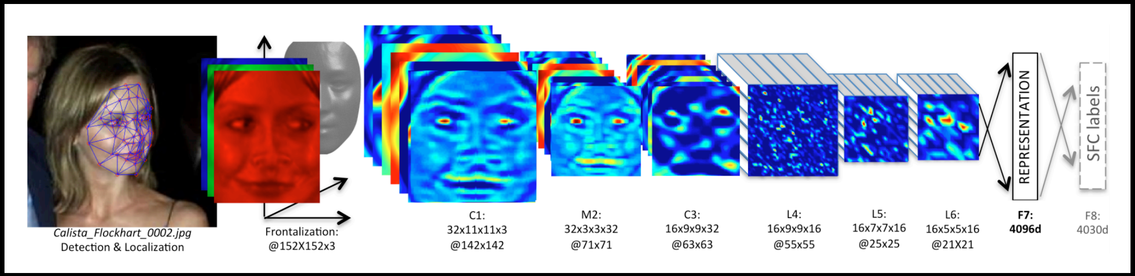


Input



Filters

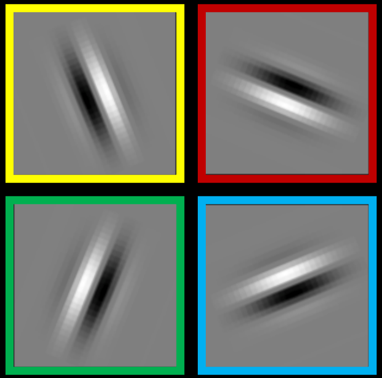
- E.g. face recognition



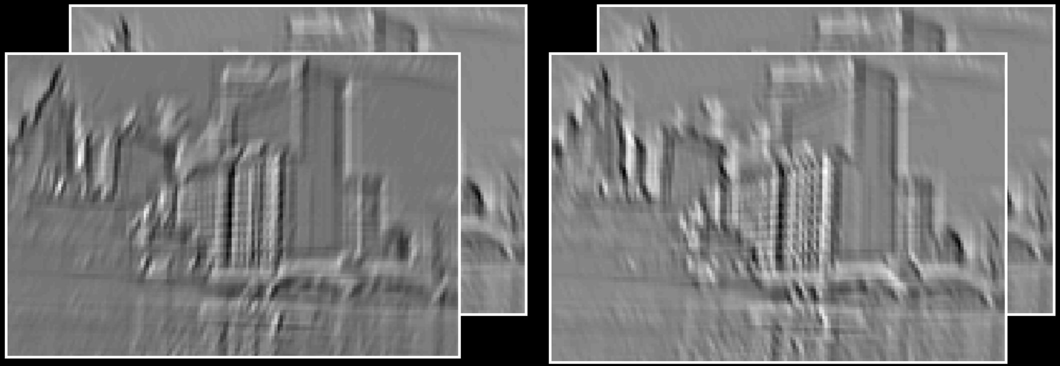
Filtering

- Tiled
  - Filters repeat every n
  - More filters than convolution for given # features

Input



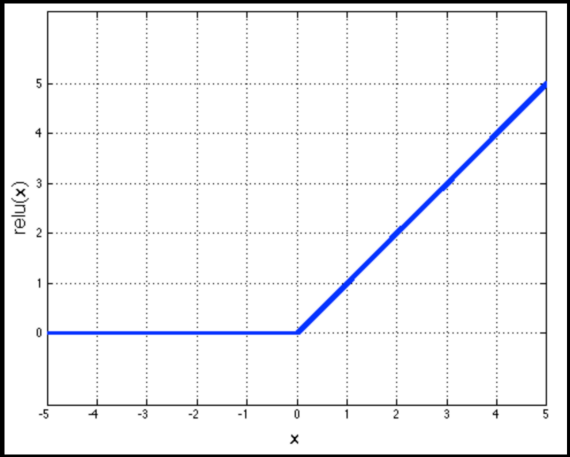
Filters



Feature maps

Non-Linearity

- Rectified linear function
  - Applied per-pixel
  - output = max(0,input)

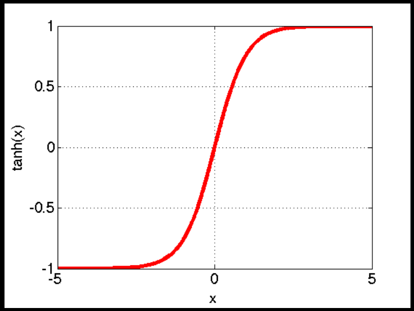


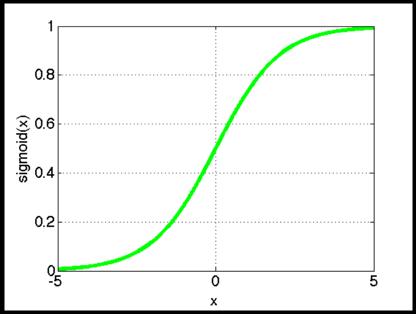
Input feature map

Output feature map

Non-Linearity

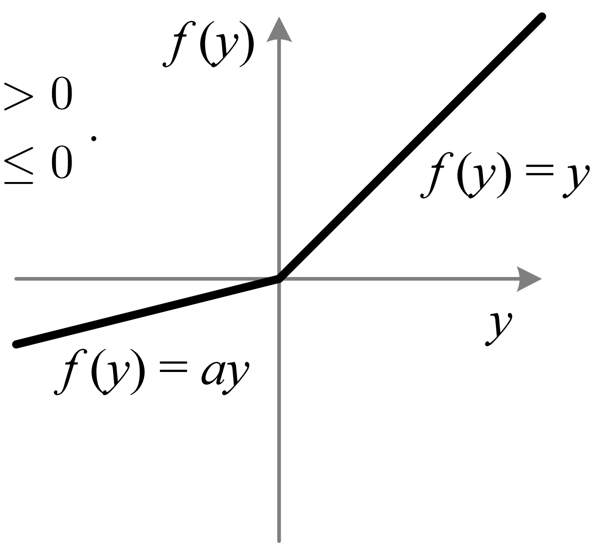
- Other choices:
  - Tanh
  - Sigmoid: 1/(1+exp(-x))
  - PReLU





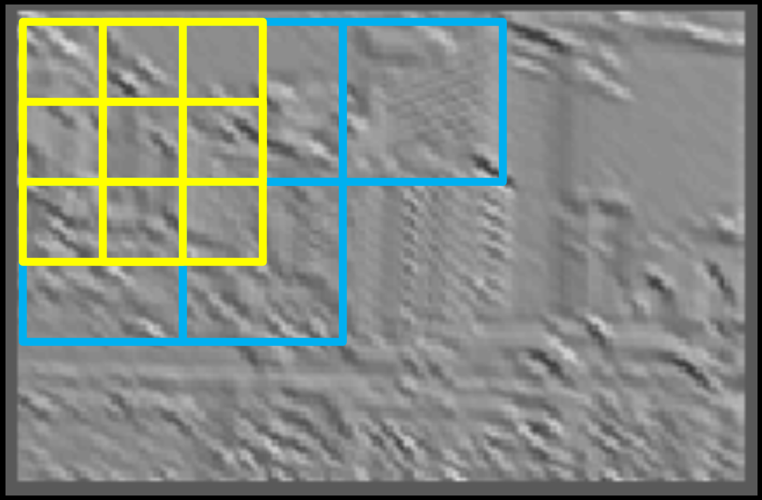
[Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification, Kaiming He et al. arXiv:1502.01852v1.pdf, Feb 2015]

$$f(y_i) = \begin{cases} y_i, & \text{if } y_i > 0 \\ a_i y_i, & \text{if } y_i \le 0 \end{cases}.$$

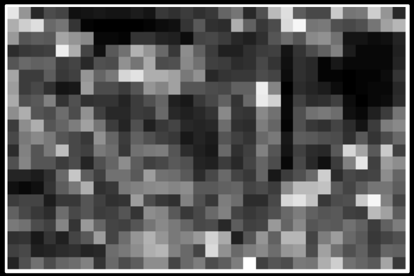


Pooling

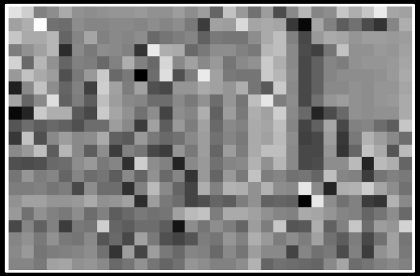
- Spatial Pooling
  - Non-overlapping / overlapping regions
  - Sum or max
  - Boureau et al. ICML'10 for theoretical analysis



Max

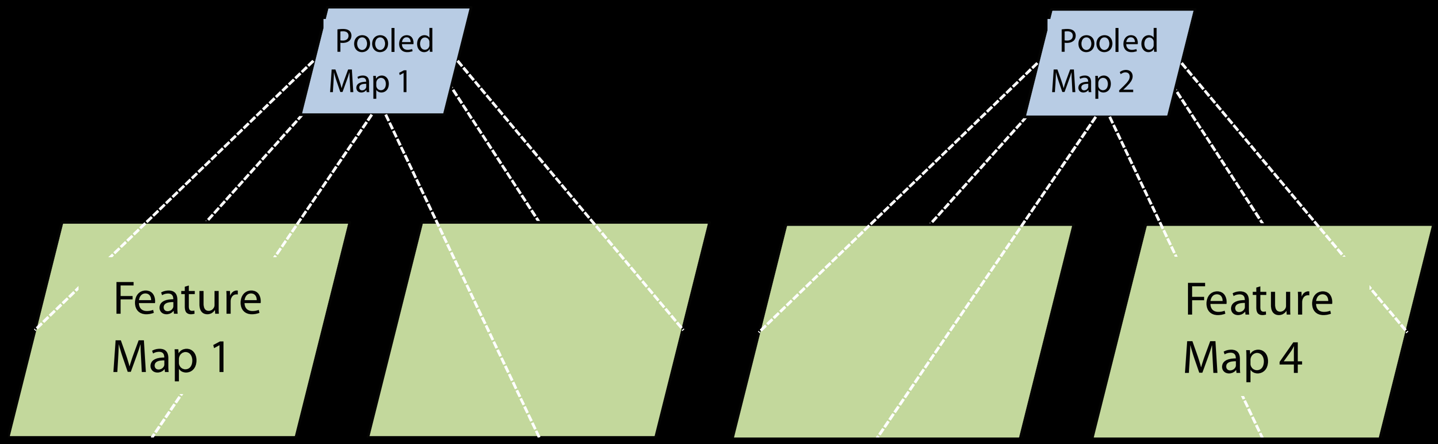


Sum



Pooling

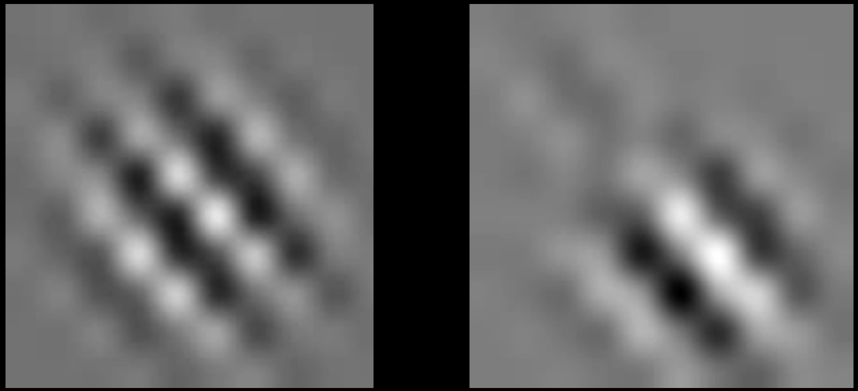
- Pooling across feature groups
  - Additional form of inter-feature competition
  - MaxOut Networks [Goodfellow et al. ICML 2013]



Role of Pooling

- Spatial pooling
  - Invariance to small transformations
  - Larger receptive fields (see more of input)

Visualization technique from [Le et al. NIPS'10]:



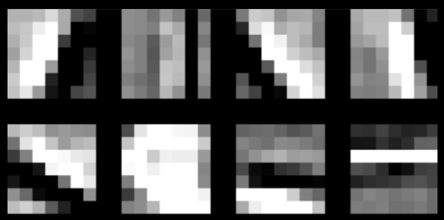
Zeiler, Fergus [arXiv 2013]

Videos from: http://ai.stanford.edu/~quocle/TCNNweb

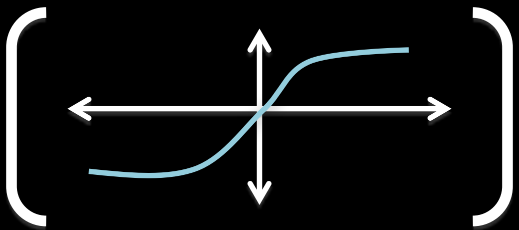
Components of Each Layer

Pixels / Features

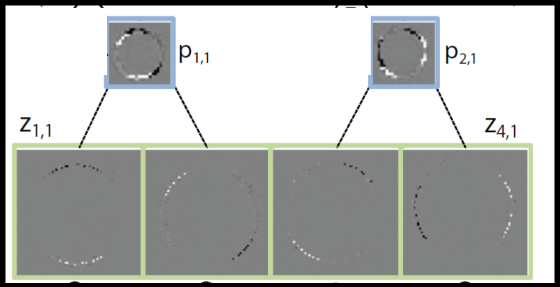
Filter with learned dictionary



Non-linearity



Spatial local max pooling



[Optional] Normalization across data/features

Output Features

Normalization

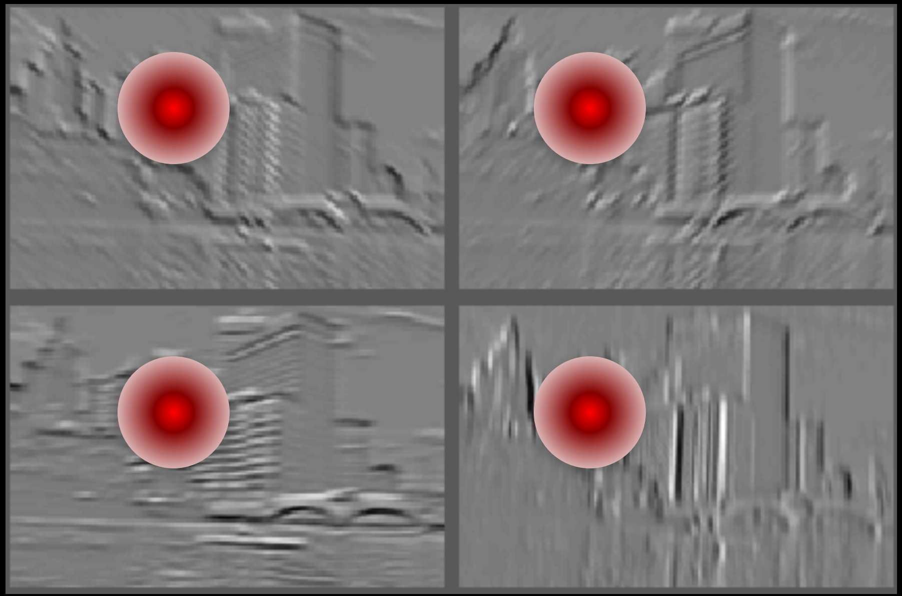
- Contrast normalization across features
  - See Divisive Normalization in Neuroscience

Input

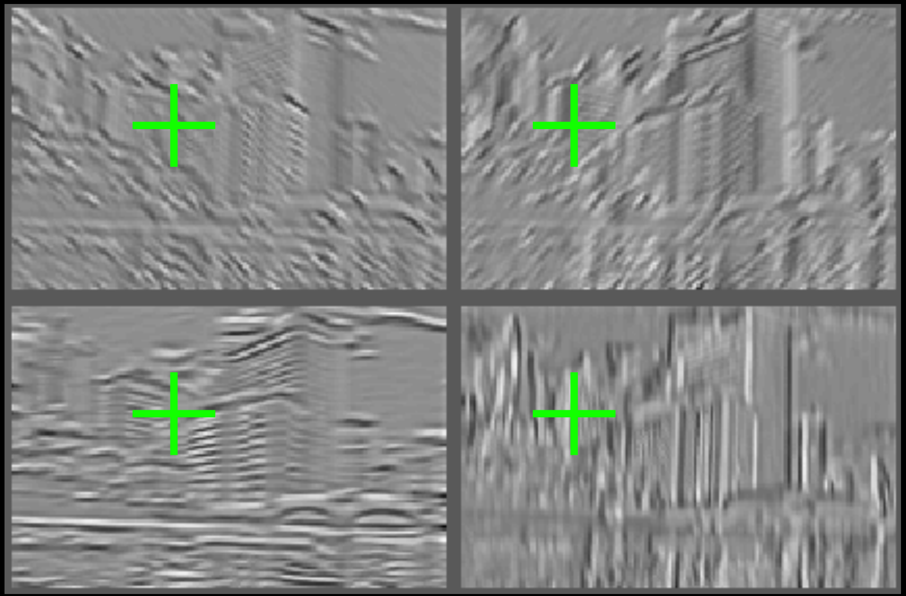
Filters

Normalization

- Contrast normalization (across feature maps)
  - Local mean = 0, local std. = 1, "Local" → 7x7 Gaussian
  - Equalizes the features maps



Feature Maps



Feature Maps
After Contrast Normalization

Role of Feature Normalization

- Introduces local competition between features
  - "Explaining away" in graphical models
  - Just like top-down models
  - But more local mechanism

- Also helps to scale activations at each layer better for learning
  - Makes energy surface more isotropic
  - So each gradient step makes more progress

- Empirically, seems to help a bit (1-2%) on ImageNet
- Most recent models don't seem to have use though

Normalization across Data

- Batch Normalization

[Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, Sergey Ioffe, Christian Szegedy, arXiv:1502.03167]

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

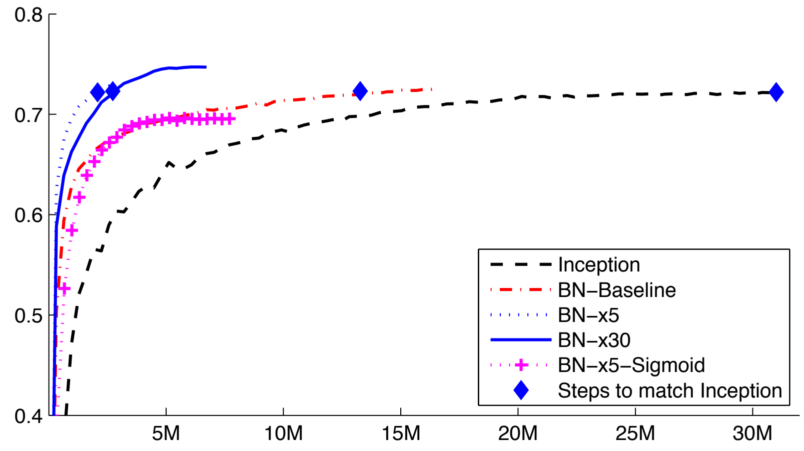
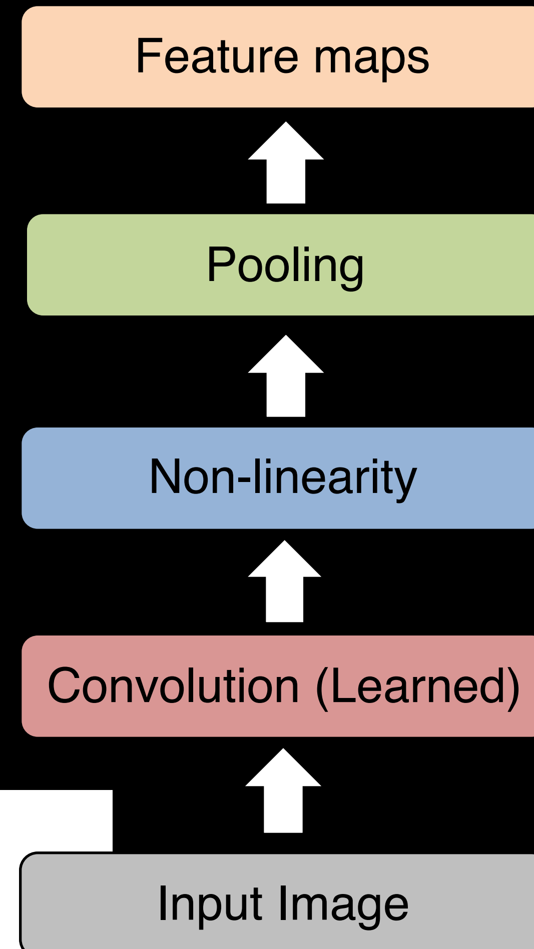
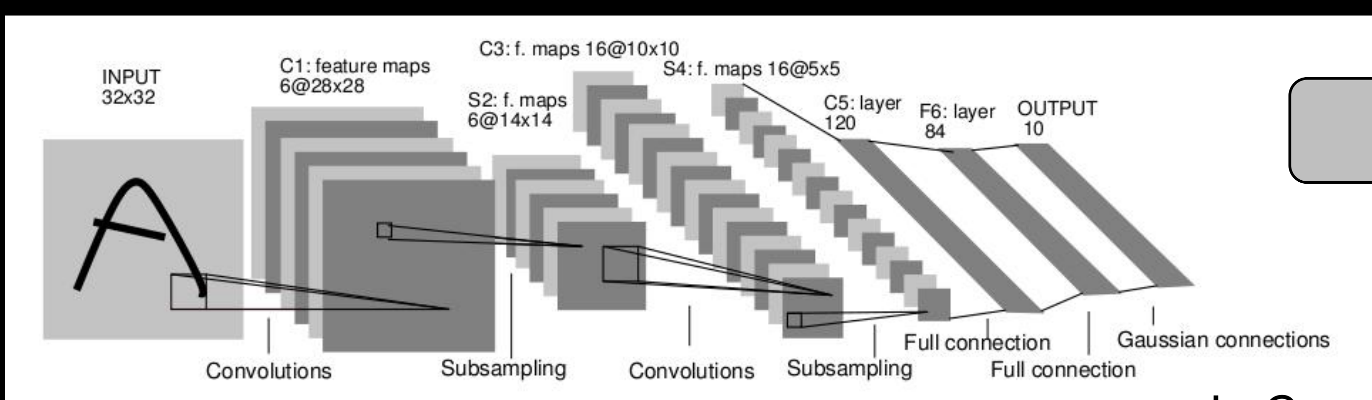


Figure 2: *Single crop validation accuracy of Inception and its batch-normalized variants, vs. the number of training steps.*

Overview of Convnets

- Feed-forward:
  - Convolve input
  - Non-linearity (rectified linear)
  - Pooling (local max)
- Supervised
- Train convolutional filters by back-propagating classification error





Architecture

- Big issue: how to select
  - Manual tuning of features → manual tuning of architechtures

- Depth
- Width
- Parameter count

How to Choose Architecture

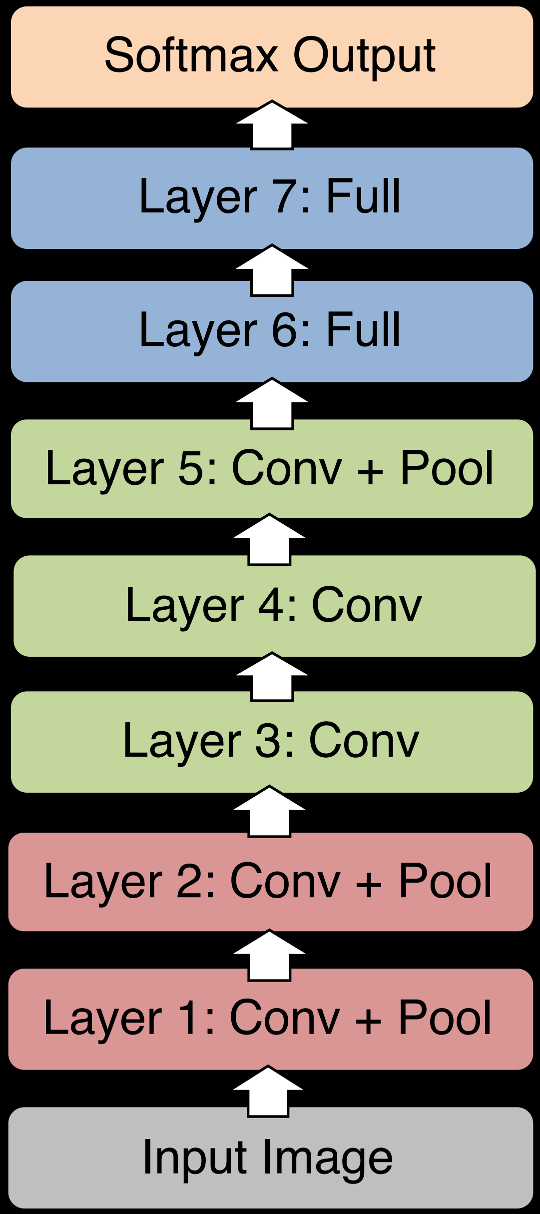
- Many hyper-parameters:
  - # layers, # feature maps
- Cross-validation
- Grid search (need lots of GPUs)
- Smarter strategies:
  - Random [Bergstra & Bengio JMLR 2012]
  - Gaussian processes [Hinton??]

How important is Depth

- “Deep” in Deep Learning
- Ablation study
- Tap off features

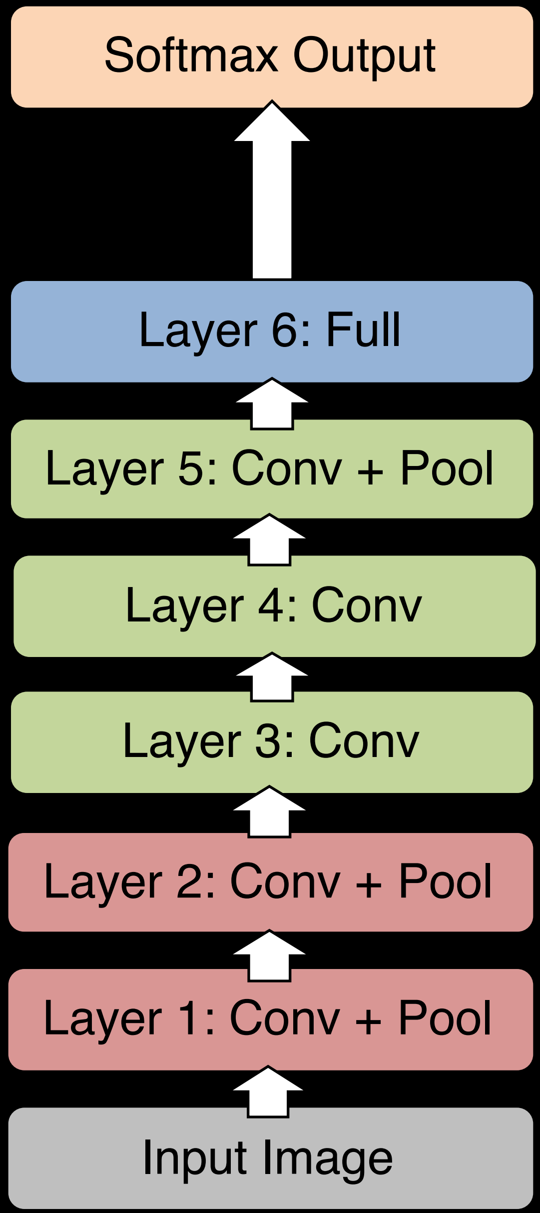
Architecture of Krizhevsky et al.

- 8 layers total
- Trained on Imagenet dataset [Deng et al. CVPR'09]
- 18.2% top-5 error
- Our reimplementation:
  18.1% top-5 error



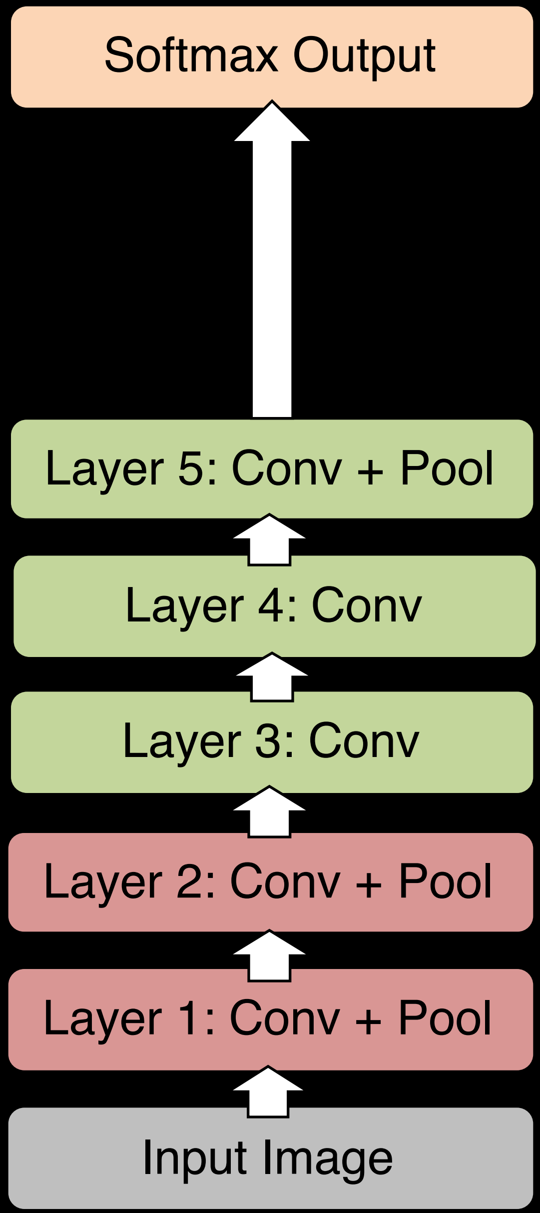
Architecture of Krizhevsky et al.

- Remove top fully connected layer
  - Layer 7
- Drop 16 million parameters
- Only 1.1% drop in performance!



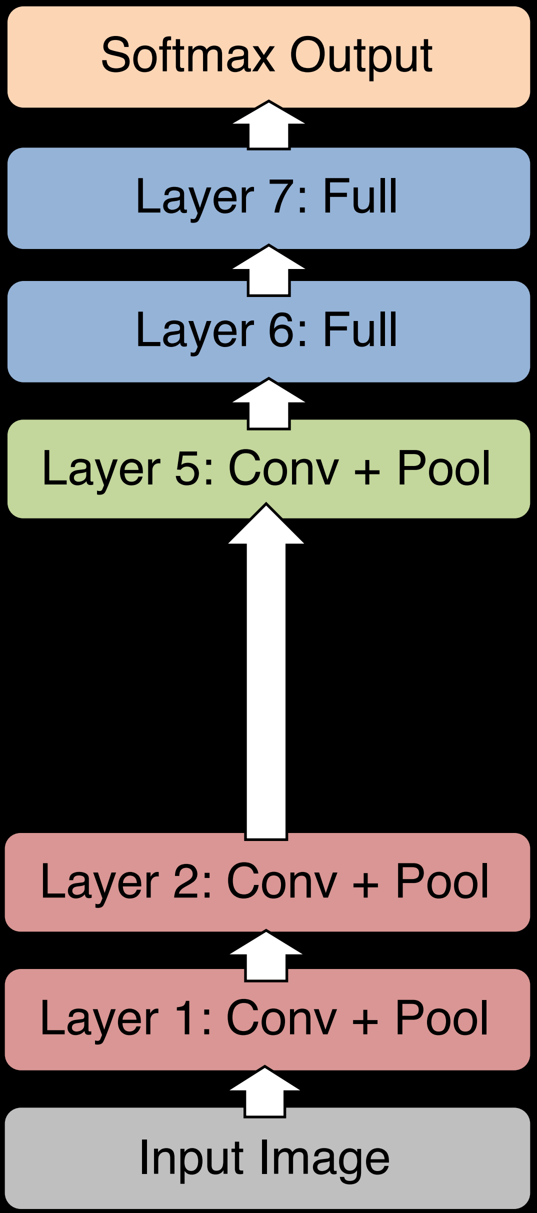
Architecture of Krizhevsky et al.

- Remove both fully connected layers
  - Layer 6 & 7
- Drop ~50 million parameters
- 5.7% drop in performance



Architecture of Krizhevsky et al.

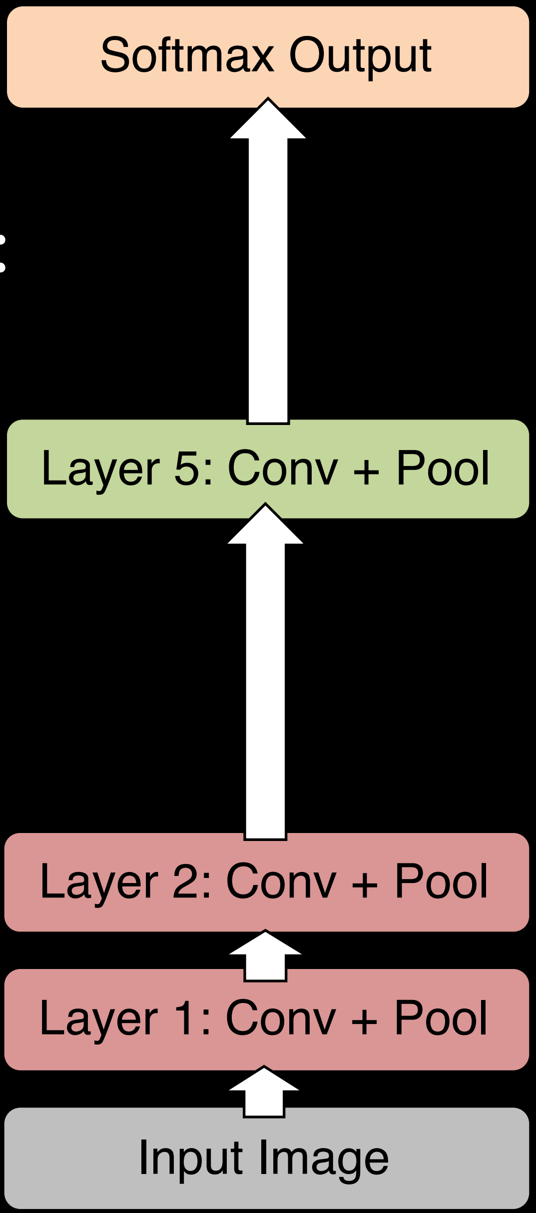
- Now try removing upper feature extractor layers:
  - Layers 3 & 4
- Drop ~1 million parameters
- 3.0% drop in performance



Architecture of Krizhevsky et al.

- Now try removing upper feature extractor layers & fully connected:
  - Layers 3, 4, 6 ,7
- Now only 4 layers
- 33.5% drop in performance

→Depth of network is key

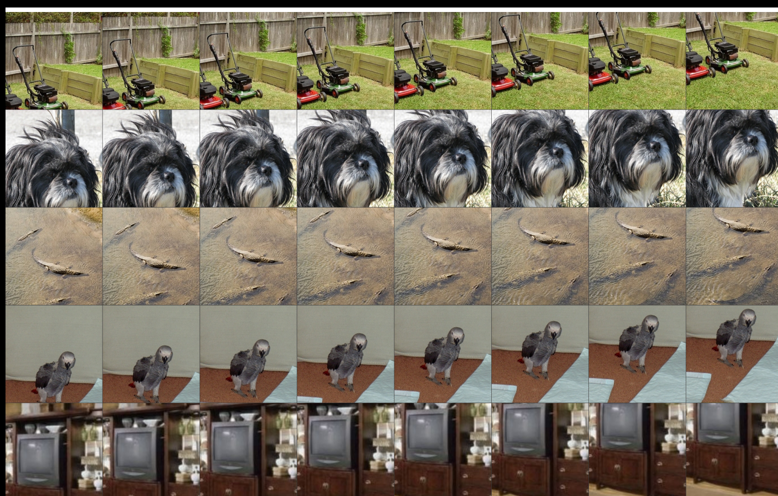


Tapping off Features at each Layer

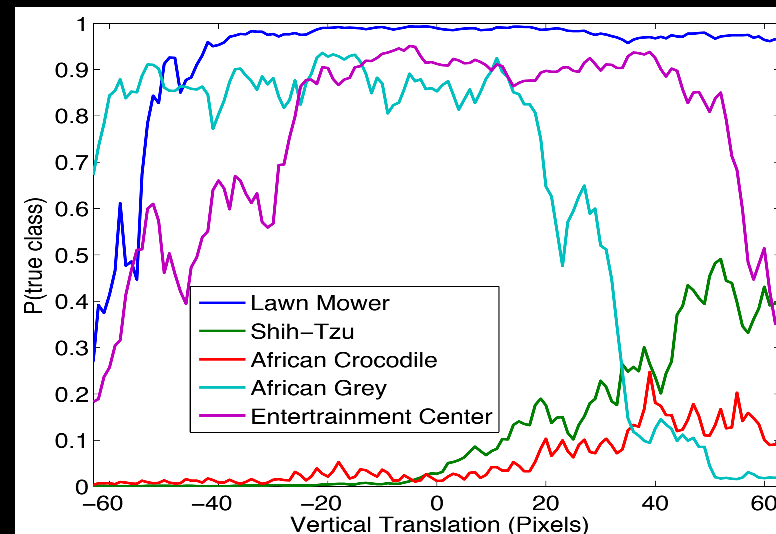
Plug features from each layer into linear SVM or soft-max

	Cal-101 (30/class)	Cal-256 (60/class)
SVM (1)	44.8 ± 0.7	24.6 ± 0.4
SVM (2)	66.2 ± 0.5	39.6 ± 0.3
SVM (3)	72.3 ± 0.4	46.0 ± 0.3
SVM (4)	76.6 ± 0.4	51.3 ± 0.1
SVM (5)	$\mathbf{86.2 \pm 0.8}$	65.6 ± 0.3
SVM (7)	$\mathbf{85.5 \pm 0.4}$	$\mathbf{71.7 \pm 0.2}$
Softmax (5)	82.9 ± 0.4	65.7 ± 0.5
Softmax (7)	$\mathbf{85.4 \pm 0.4}$	$\mathbf{72.6 \pm 0.1}$

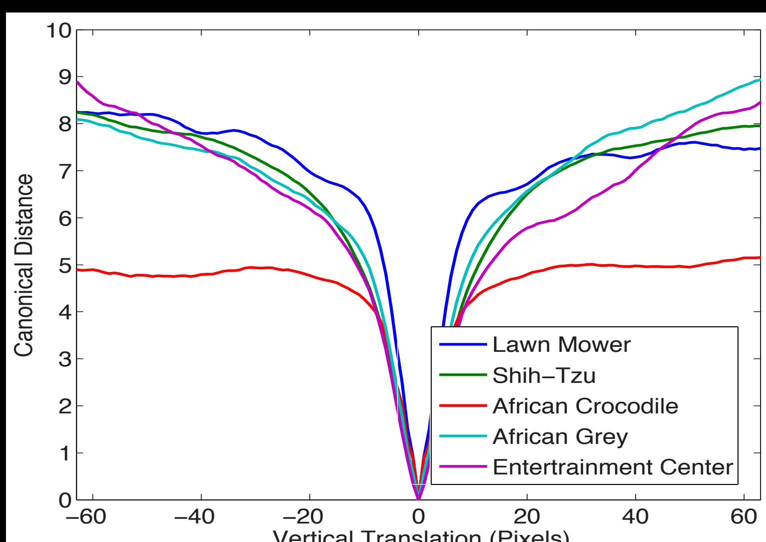
Translation (Vertical)



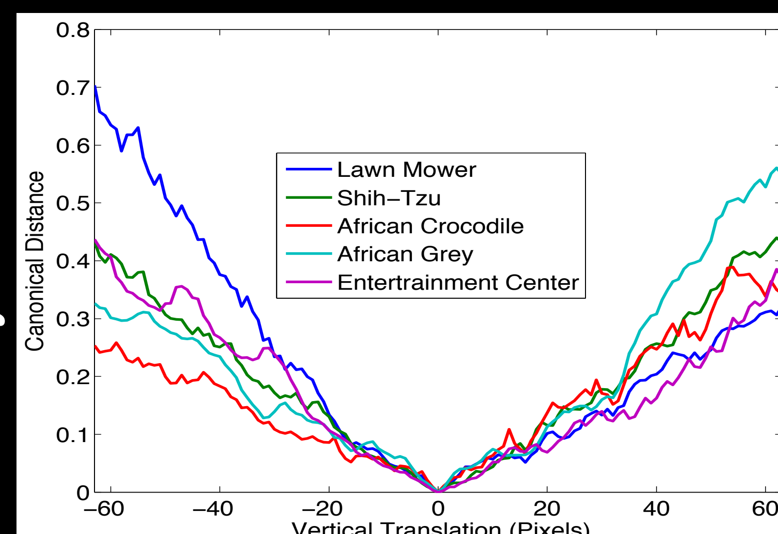
Output



Layer 1

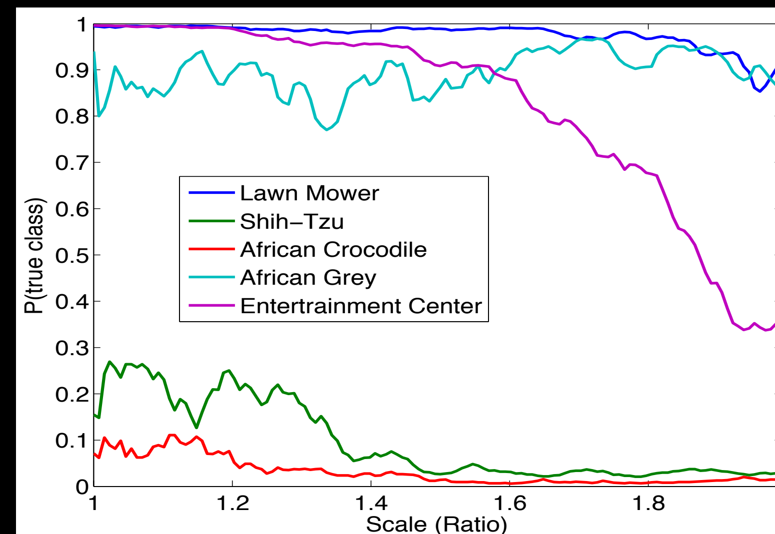


Layer 7

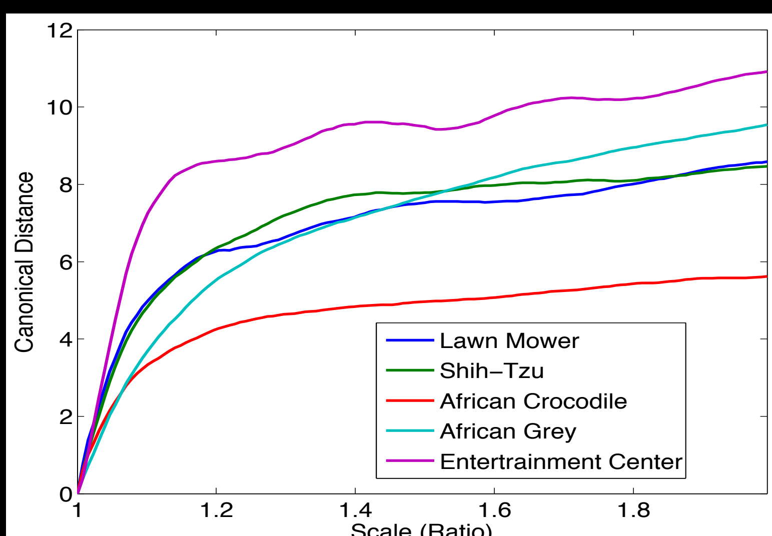


Scale Invariance

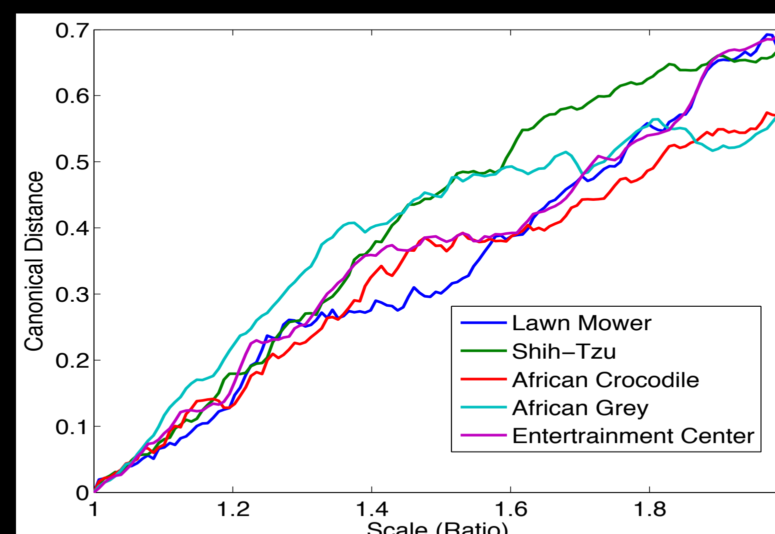
Output



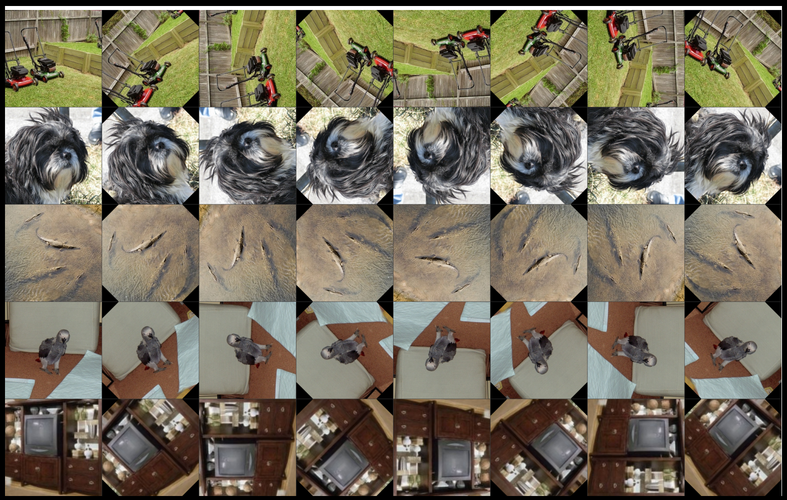
Layer 1



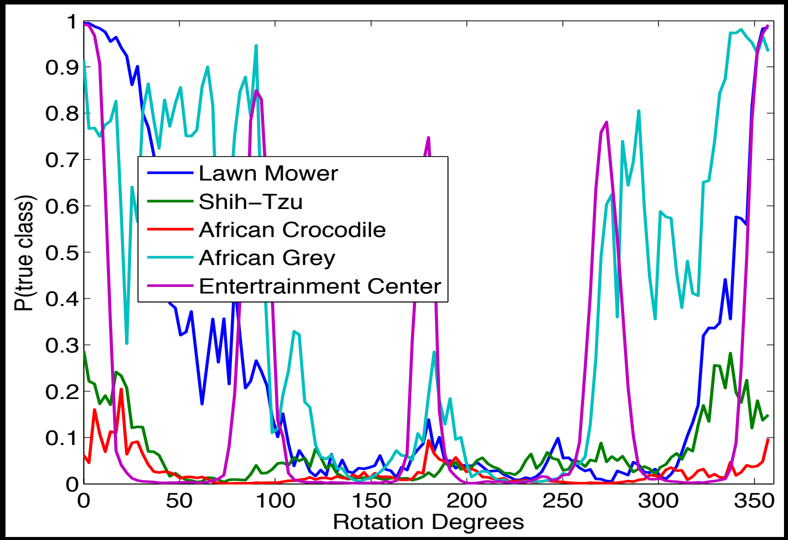
Layer 7



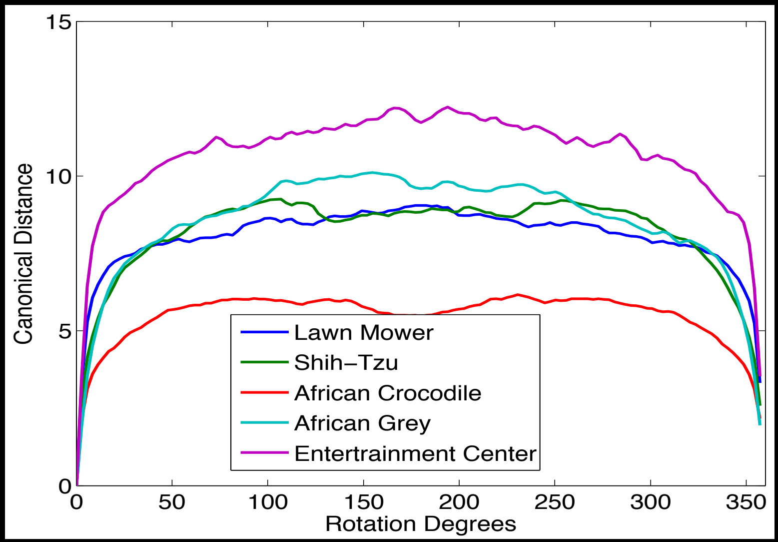
Rotation Invariance



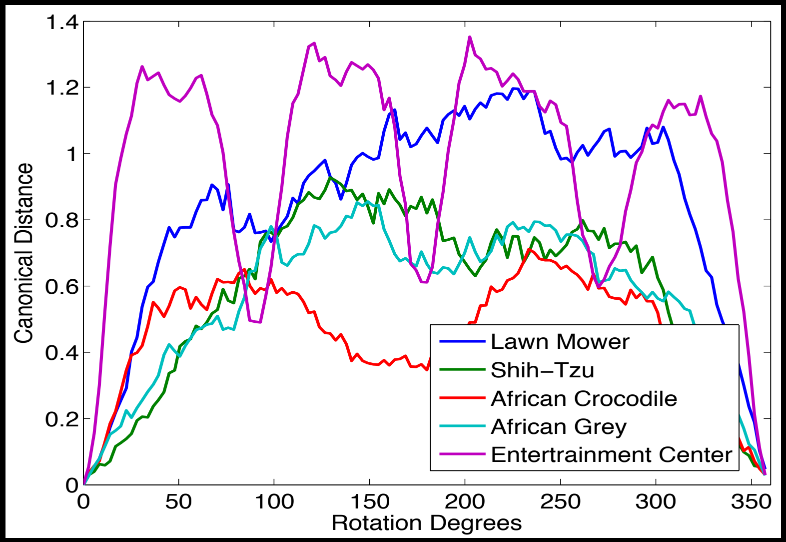
Output



Layer 1



Layer 7



Very Deep Models (1)

[Very Deep Convolutional Networks for Large-Scale Image Recognition, Karen Simonyan & Andrew Zisserman, arXiv:1409.1556, 2014]

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

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

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

- Lots of 3x3 conv layers: more non-linearity than single 7x7 layer
- Close to SOA results on Imagenet: 6.8% top-5 val
- Can be hard to train

Table 3: **ConvNet performance at a single test scale.**

ConvNet config. (Table 1)	smallest image side train (S)	smallest image side test (Q)	top-1 val. error (%)	top-5 val. error (%)
A	256	256	29.6	10.4
A-LRN	256	256	29.7	10.5
B	256	256	28.7	9.9
C	256	256	28.1	9.4
	384	384	28.1	9.3
	[256;512]	384	27.3	8.8
D	256	256	27.0	8.8
	384	384	26.8	8.7
	[256;512]	384	25.6	8.1
E	256	256	27.3	9.0
	384	384	26.9	8.7
	[256;512]	384	**25.5**	**8.0**

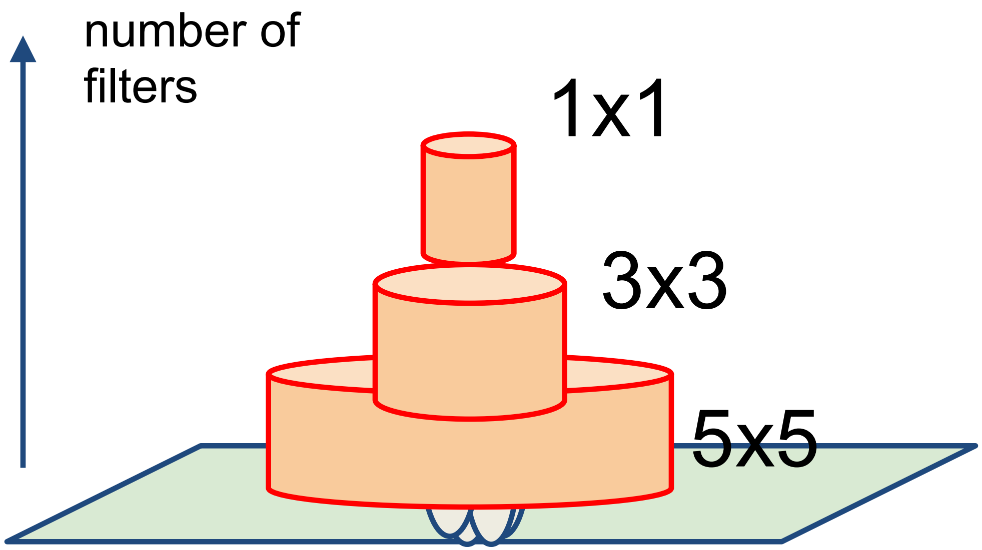
Very Deep Models (2)

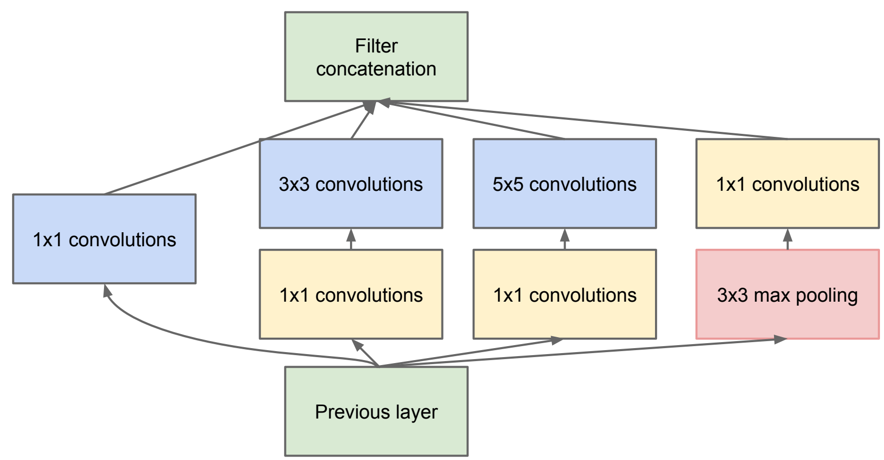
[Going Deep with Convolutions, Szegedy et al., arXiv:1409.4842, 2014]

GoogLeNet inception module:

1. Multiple filter scales at each layer

2. Dimensionality reduction to keep computational requirements down

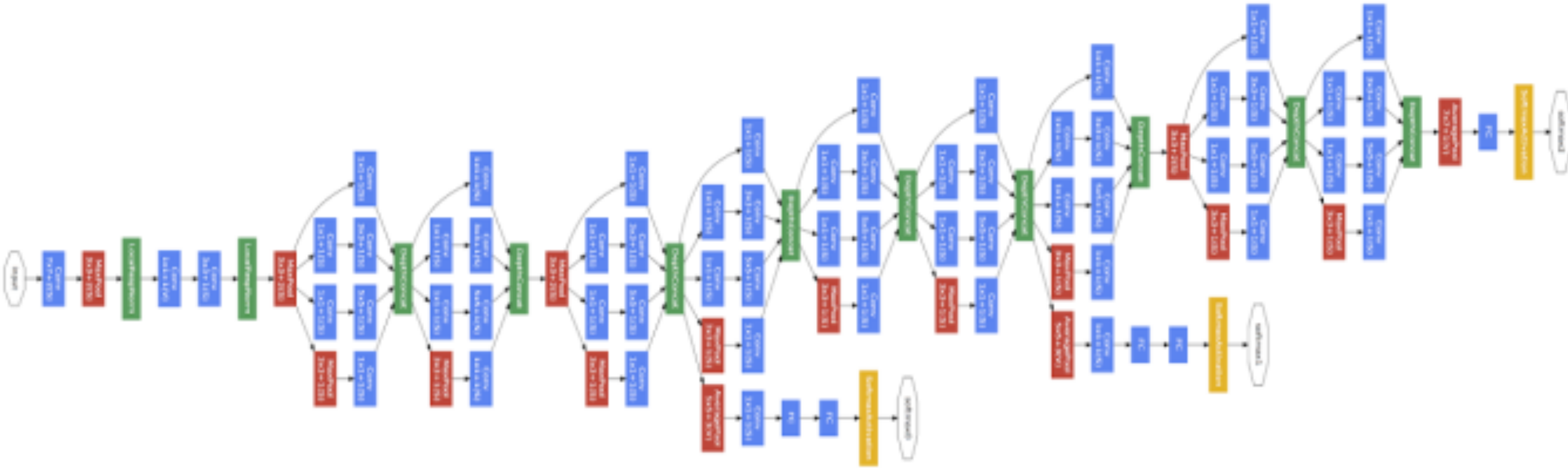




[From http://image-
net.org/challenges/LSVRC/2014/slides/Go

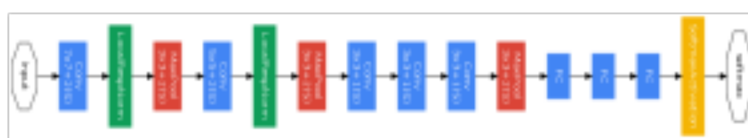
GoogLeNet vs Previous Models

[Going Deep with Convolutions, Szegedy et al., arXiv:1409.4842, 2014]



GoogLeNet

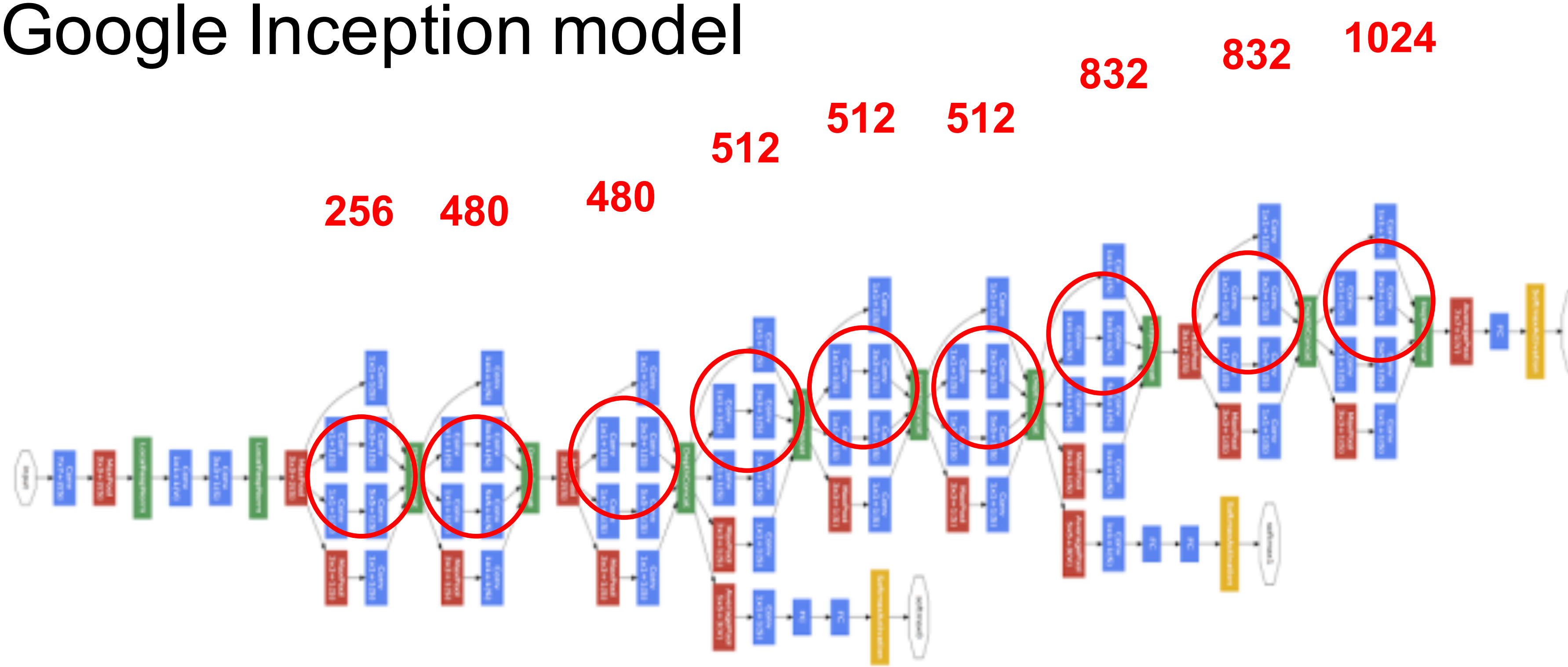
Convolution
Pooling
Softmax
Other



Zeiler-Fergus Architecture (1 tower)

[From http://image-net.org/challenges/LSVRC/2014/slides

Google Inception model



Width of inception modules ranges from 256 filters (in early modules) to 1024 in top inception modules.

Can remove fully connected layers on top completely

Number of parameters is reduced to 5 million

6.7% top-5 validation error on Imagnet

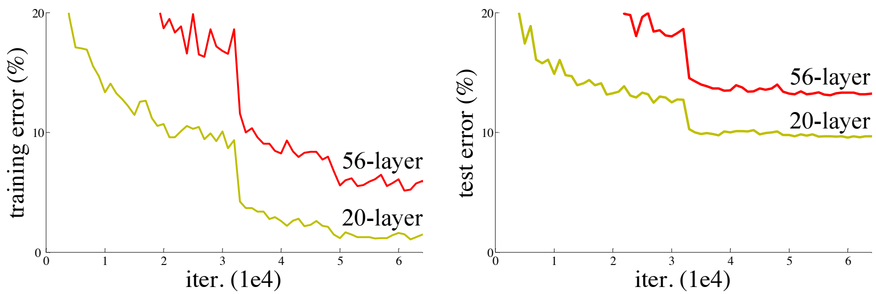
Computional cost is increased by less than 2X compared to Krizhevsky's network. (<1.5Bn operations/evaluation)

[From http://image-net.org/challenges/LSVRC/2014/slides/Go

Residual Networks

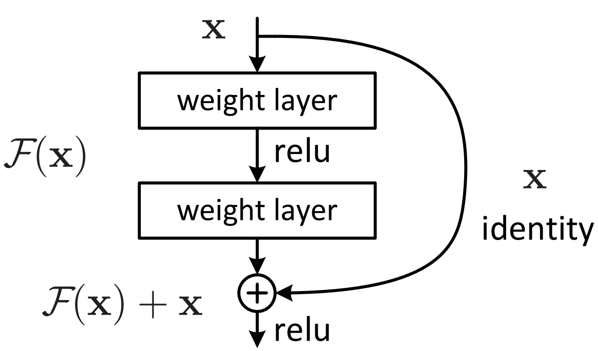
[He, Zhang, Ren, Sun, CVPR 2016]

Really, really deep convnets don't train well,
E.g. CIFAR10:



Key idea: introduce "pass through" into each layer

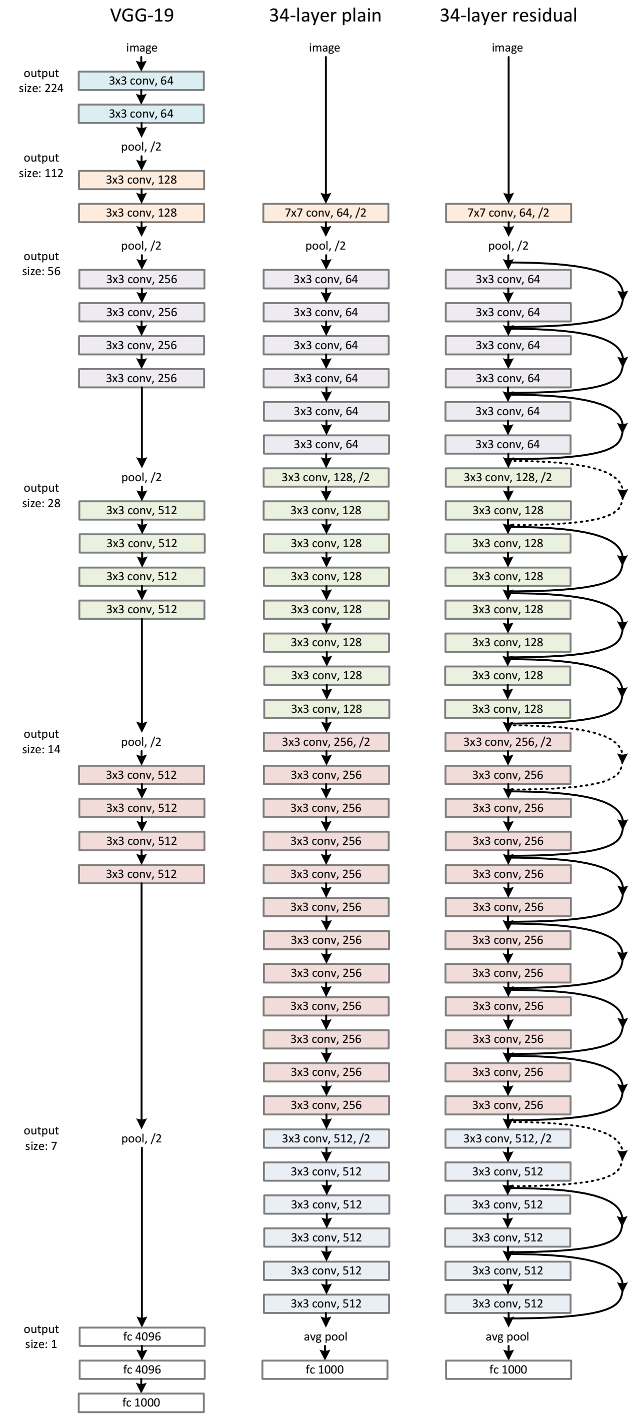
Thus only residual now needs to be learned



method	top-1 err.	top-5 err.
VGG [41] (ILSVRC'14)	-	8.43†
GoogLeNet [44] (ILSVRC'14)	-	7.89
VGG [41] (v5)	24.4	7.1
PReLU-net [13]	21.59	5.71
BN-inception [16]	21.99	5.81
ResNet-34 B	21.84	5.71
ResNet-34 C	21.53	5.60
ResNet-50	20.74	5.25
ResNet-101	19.87	4.60
ResNet-152	**19.38**	**4.49**

Table 4. Error rates (%) of **single-model** results on the ImageNet validation set (except † reported on the test set).

With ensembling, 3.57% top-5 test error on ImageNet



Visualizing Convnets

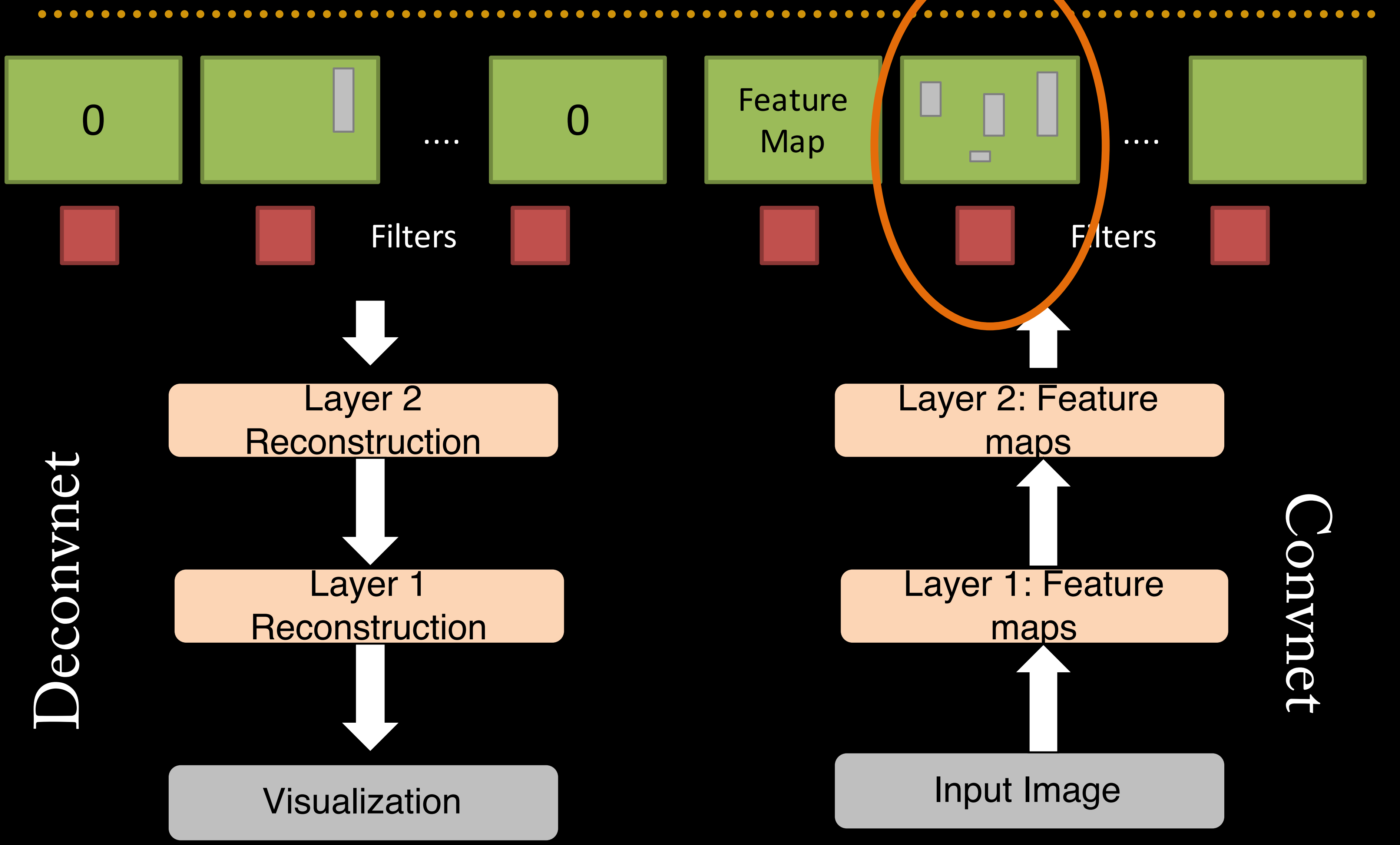
- Want to know what they are learning
- Raw coefficients of learned filters in higher layers difficult to interpret
- Two classes of method:
  1. Project activations back to pixel space
  2. Optimize input image to maximize a particular feature map or class

Visualizing Convnets

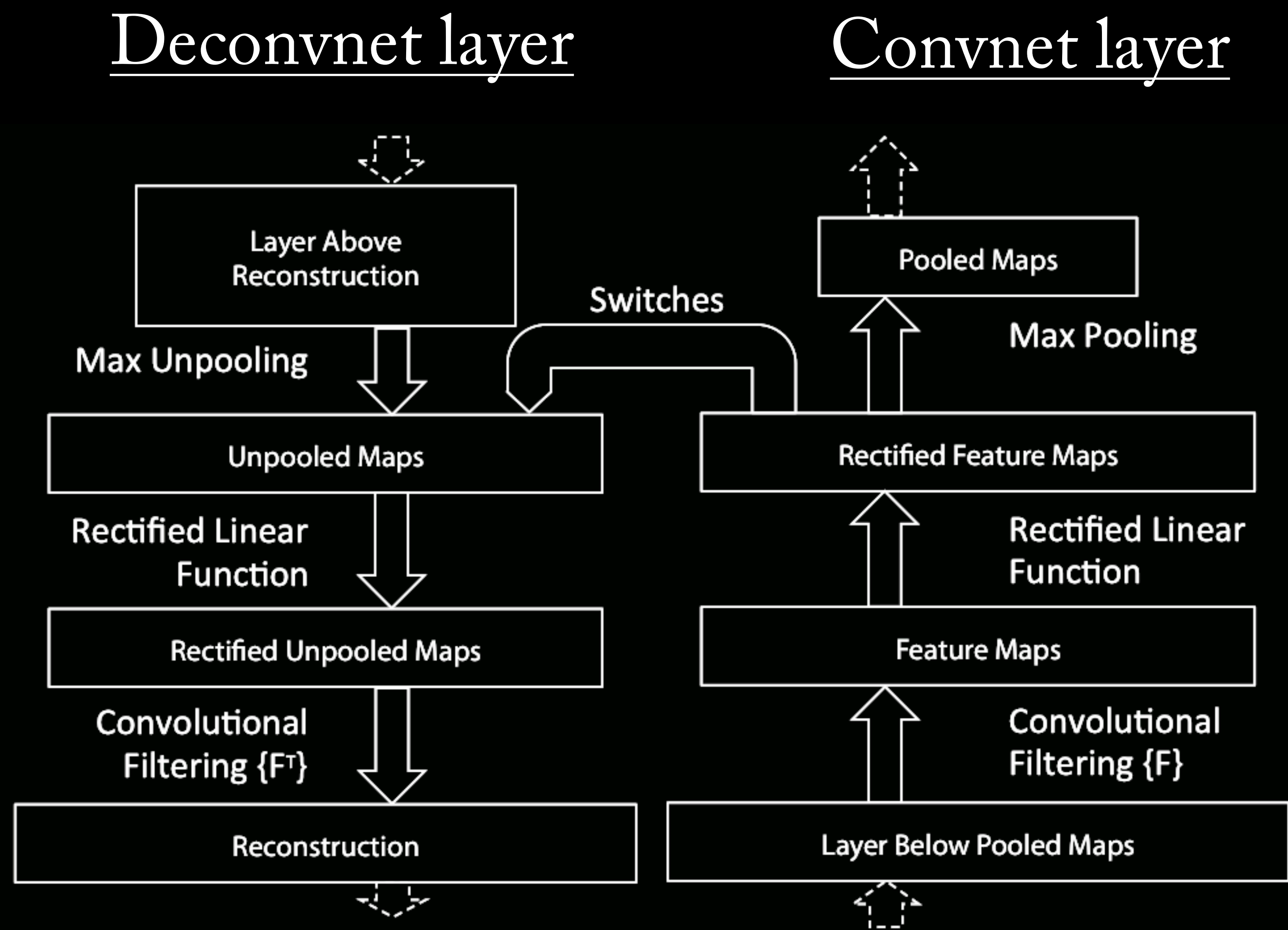
- Projection from higher layers back to input
  - Several similar approaches:
  - Visualizing and Understanding Convolutional Networks, Matt Zeiler & Rob Fergus, ECCV 2014
  - Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, Karen Simonyan, Andrea Vedaldi, Andrew Zisserman, arXiv 1312.6034, 2013
  - Object Detectors Emerge in Deep Scene CNNs, Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, Antonio Torralba, ICLR 2015

Projection from Higher Layers

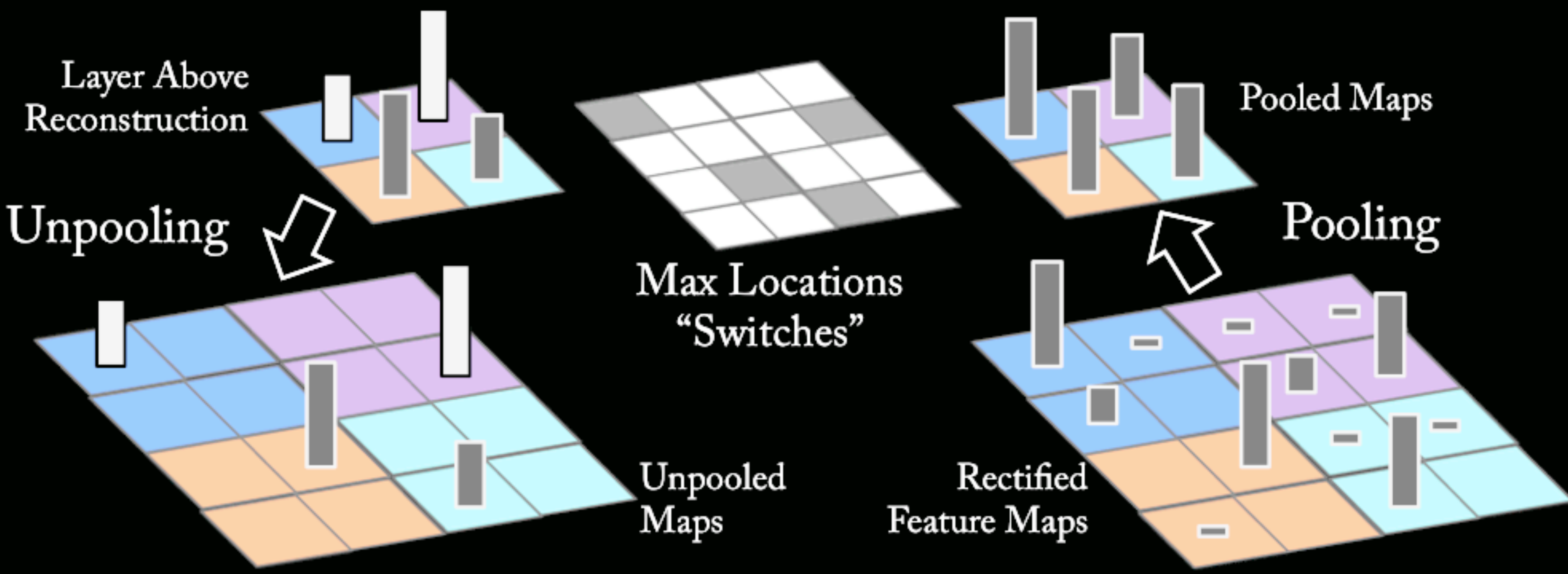
[Zeiler et al. ECCV14]



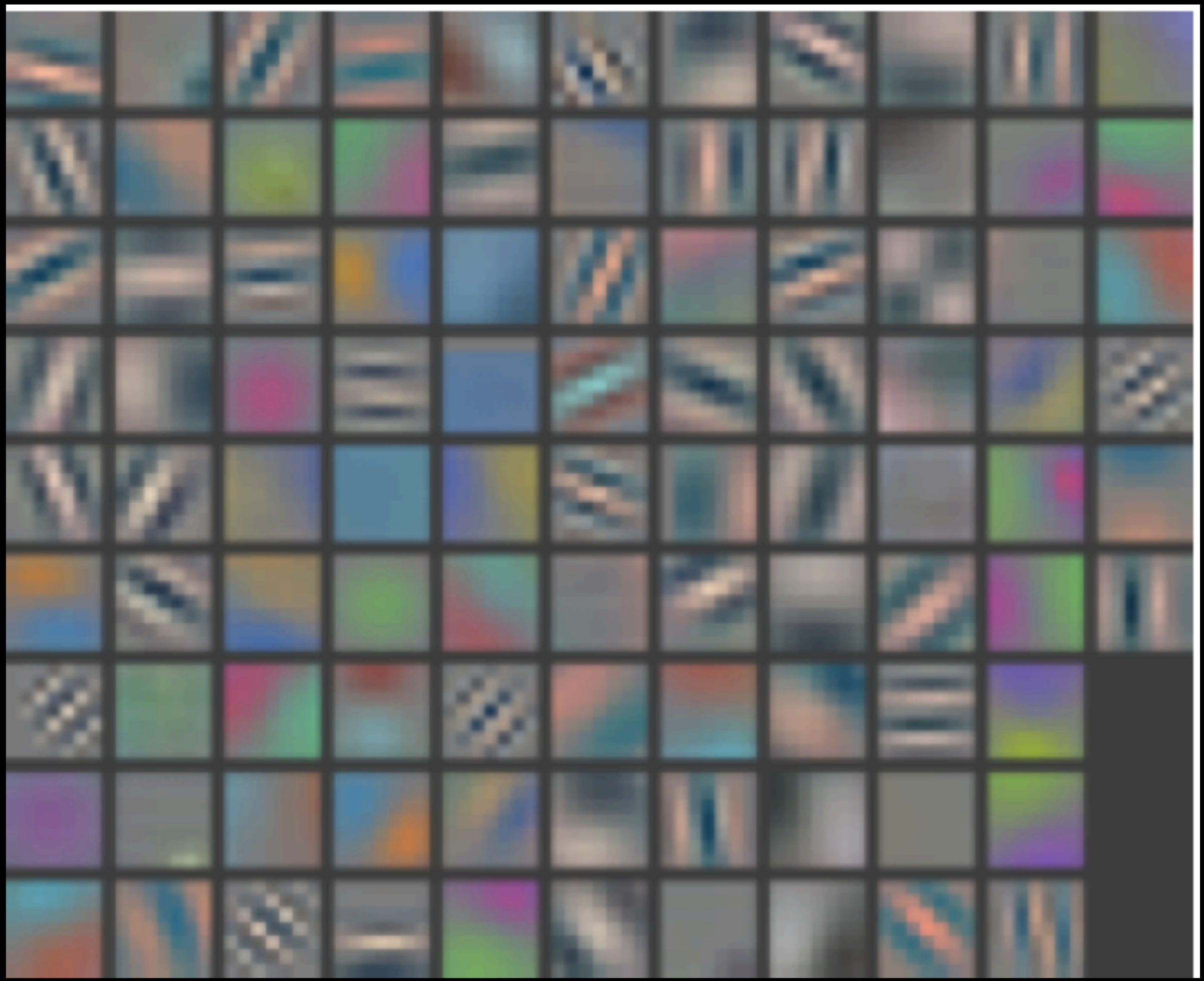
Details of Operation



Unpooling Operation

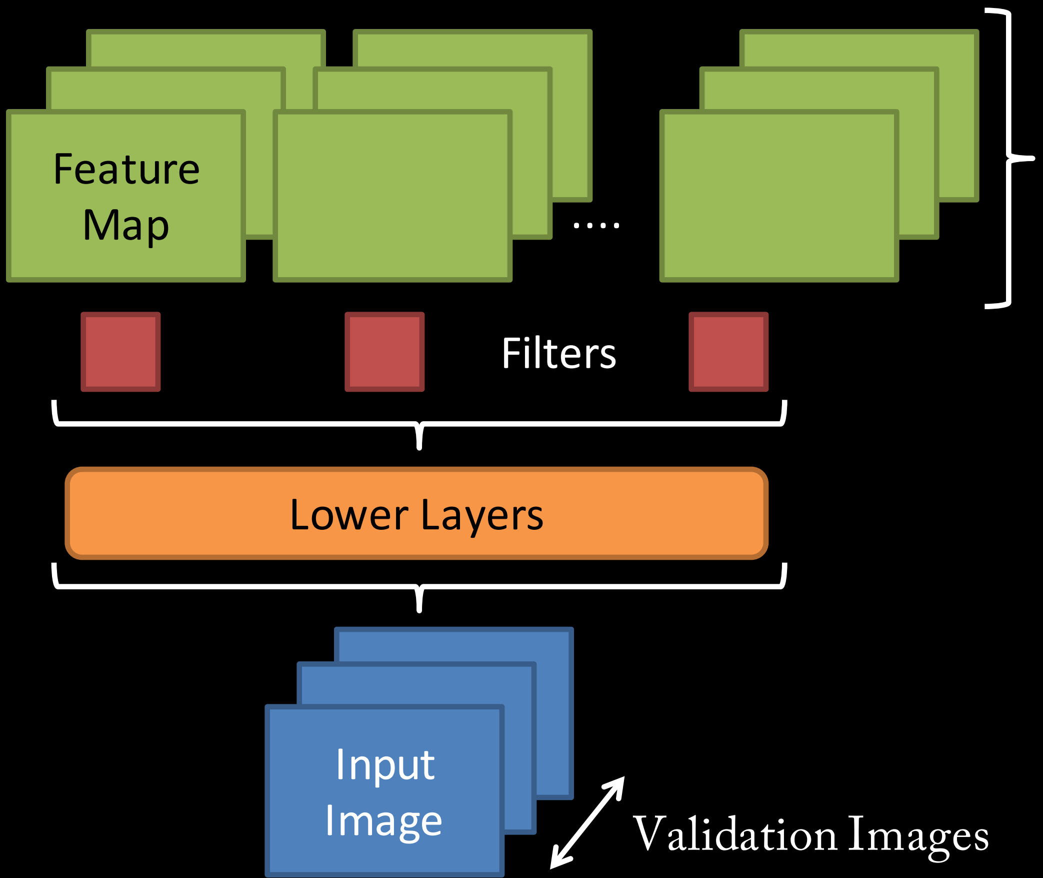


Layer 1 Filters



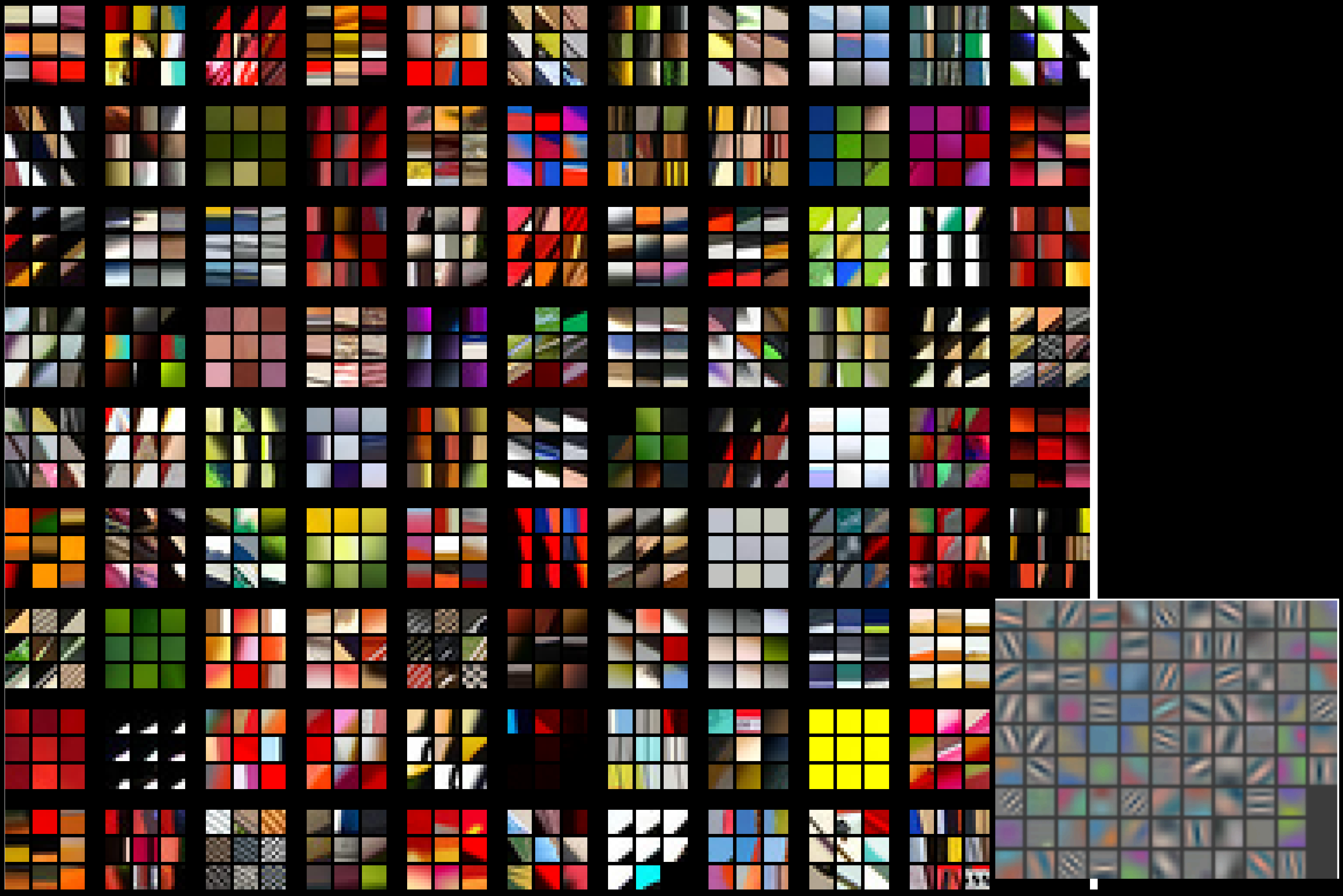
Visualizations of Higher Layers

- Use ImageNet 2012 validation set
- Push each image through network

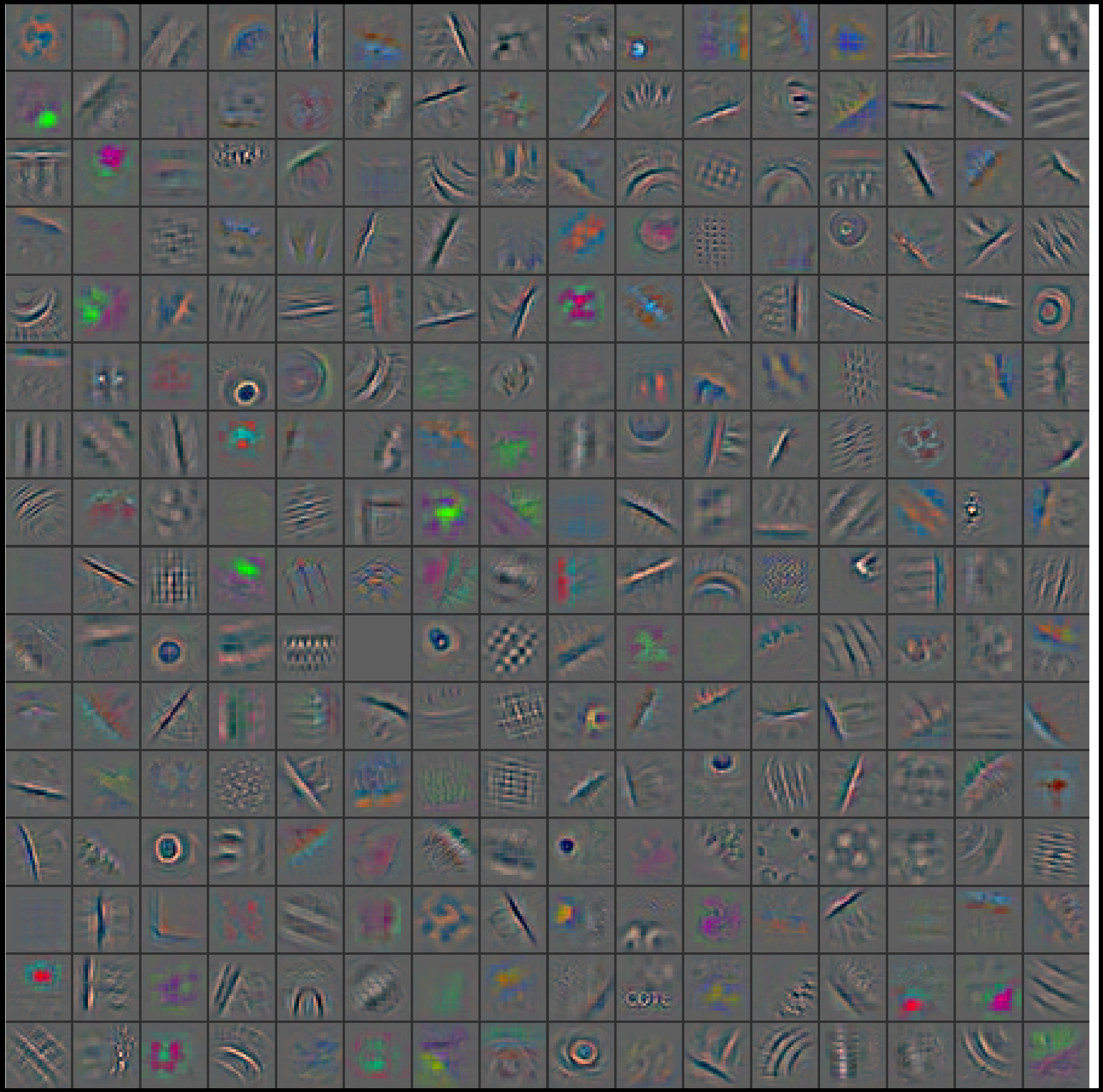


- Take max activation from feature map associated with each filter
- Use Deconvnet to project back to pixel space
- Use pooling "switches" peculiar to that activation

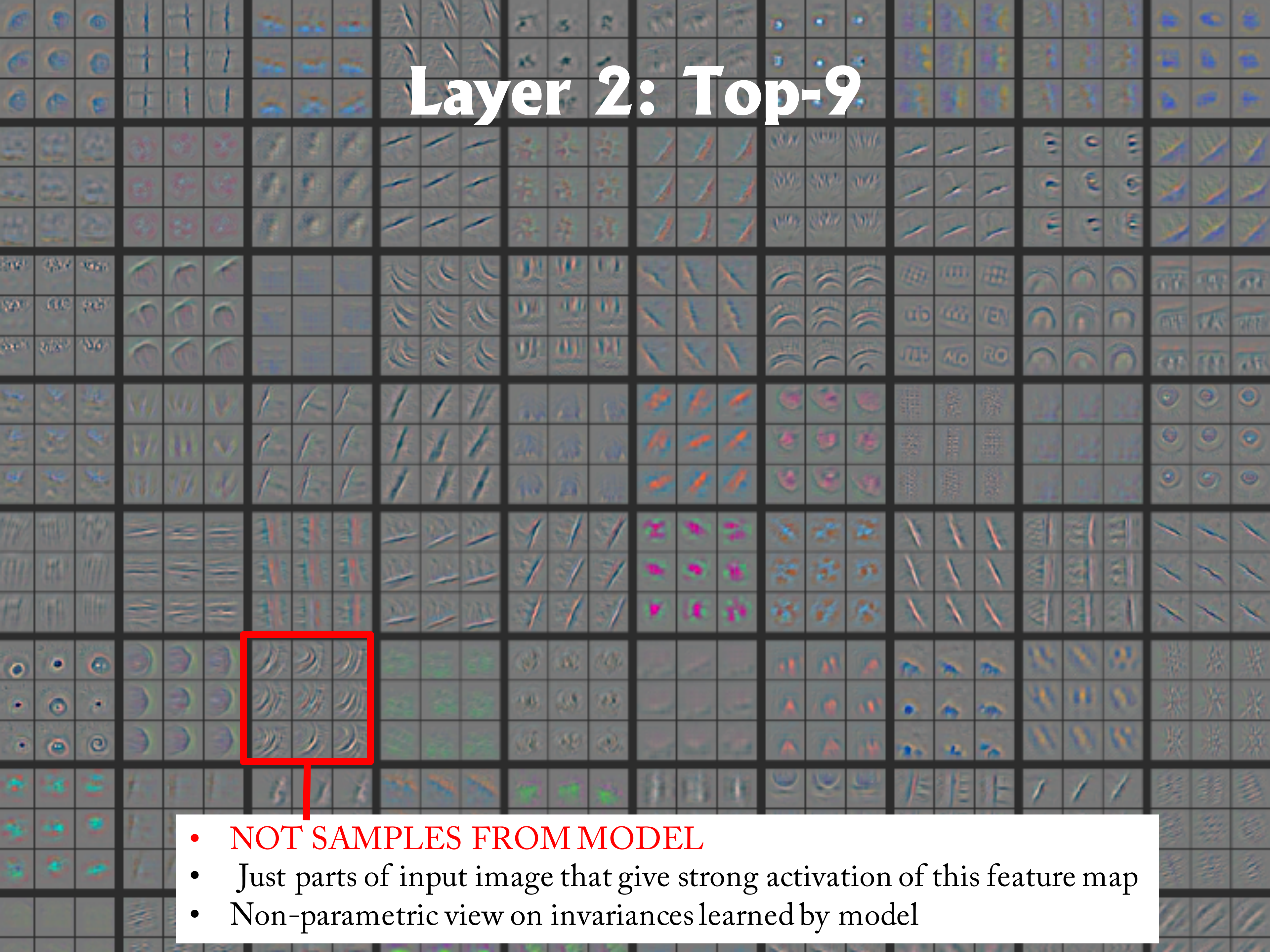
Layer 1: Top-9 Patches



Layer 2: Top-1



Layer 2: Top-9



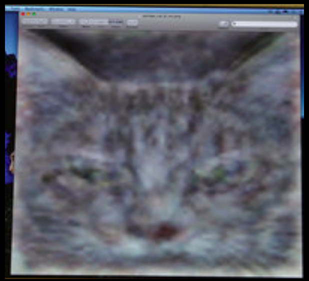
- NOT SAMPLES FROM MODEL
- Just parts of input image that give strong activation of this feature map
- Non-parametric view on invariances learned by model

Layer 2: Top-9 Patches

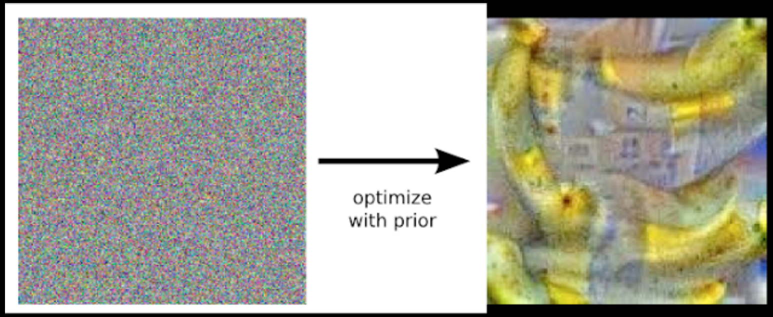
- Patches from validation images that give maximal activation of a given feature map

Visualizing Convnets

- Optimize input to maximize particular ouput
  - Lots of approaches, e.g. Erhan et al. [Tech Report 2009], Le et al. [NIPS 2010].
  - Depend on initialization



- Google DeepDream [http://googleresearch.blogspot.ch/2015/06/inceptionism-going-deeper-into-neural.html]
  - Maximize "banana" output



Google DeepDream

https://photos.google.com/share/F1QipPX0SCl7OzWilt9LnuQliattX4OUCj_8EP65_cTVnBmS1jnYgsGQAieQUc1VQWd
gQ/photo/AF1QipMYTXpt0TvZ0Q5kubkGw8VAq2isxBuL02wKZafB?key=aVBxWjhwSzg2RjJWLWRuVFBBZEN1d20
5bUdEMnhB

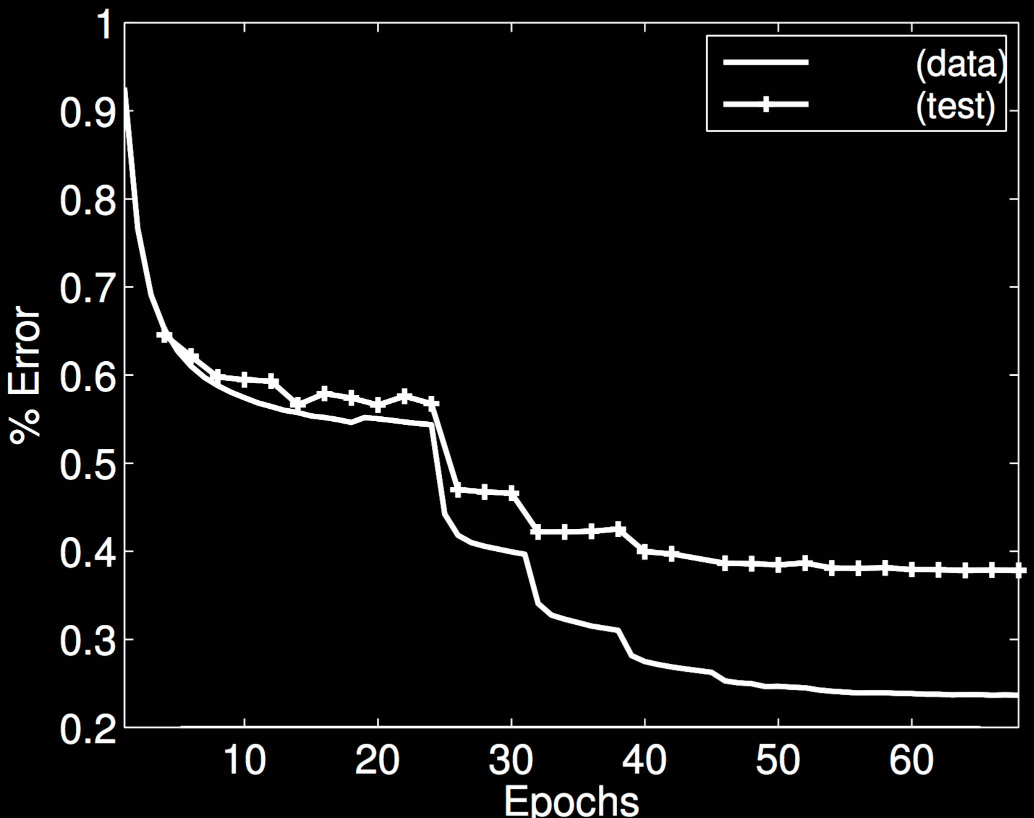
Training Big ConvNets

- Stochastic Gradient Descent
  - Compute (noisy estimate of) gradient on small batch of data & make step
  - Take as many steps as possible (even if they are noisy)
  - Large initial learning rate
  - Anneal learning rate

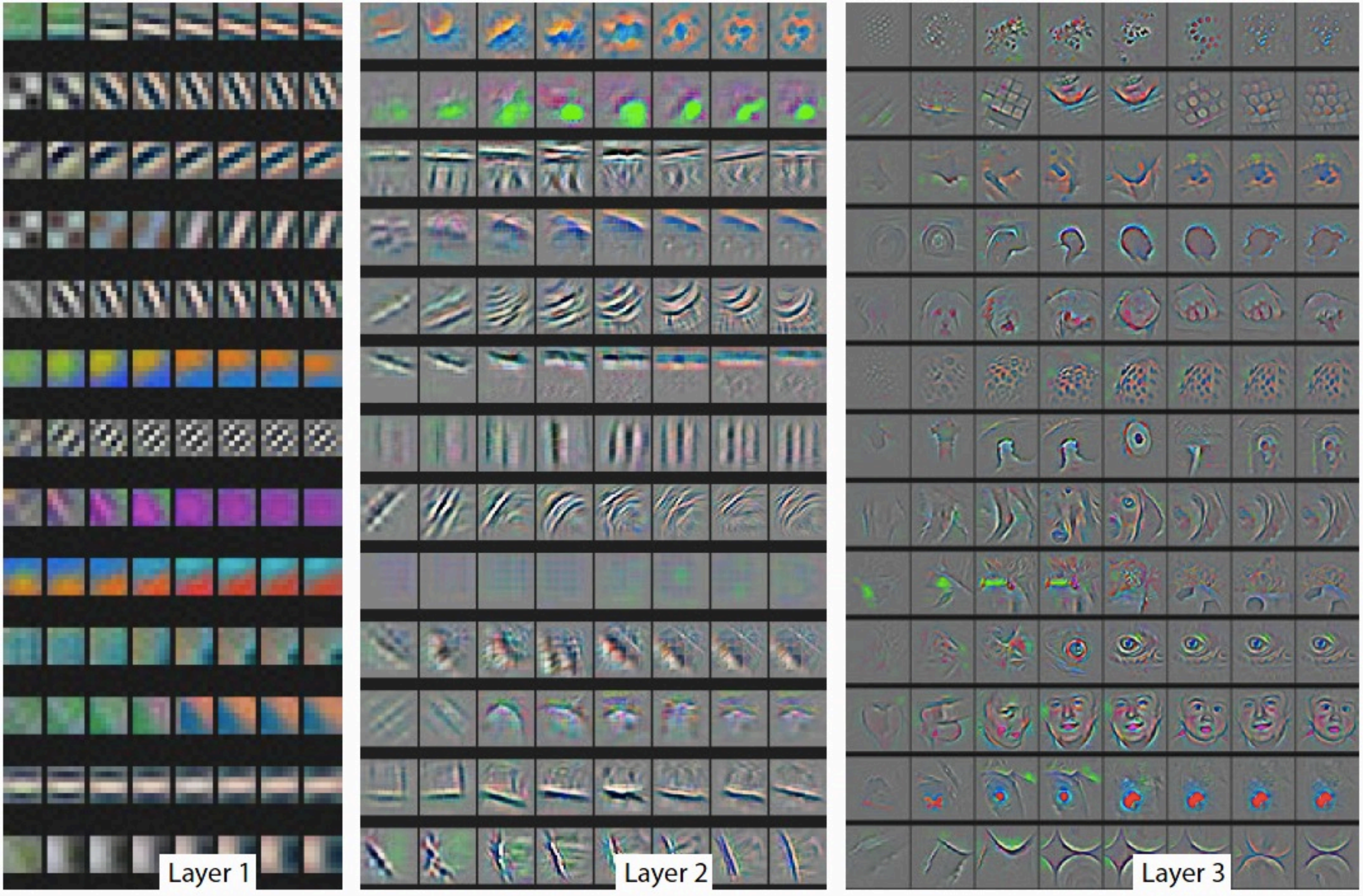
- Momentum
  - Variants [Sutskever ICML 2012]

Annealing of Learning Rate

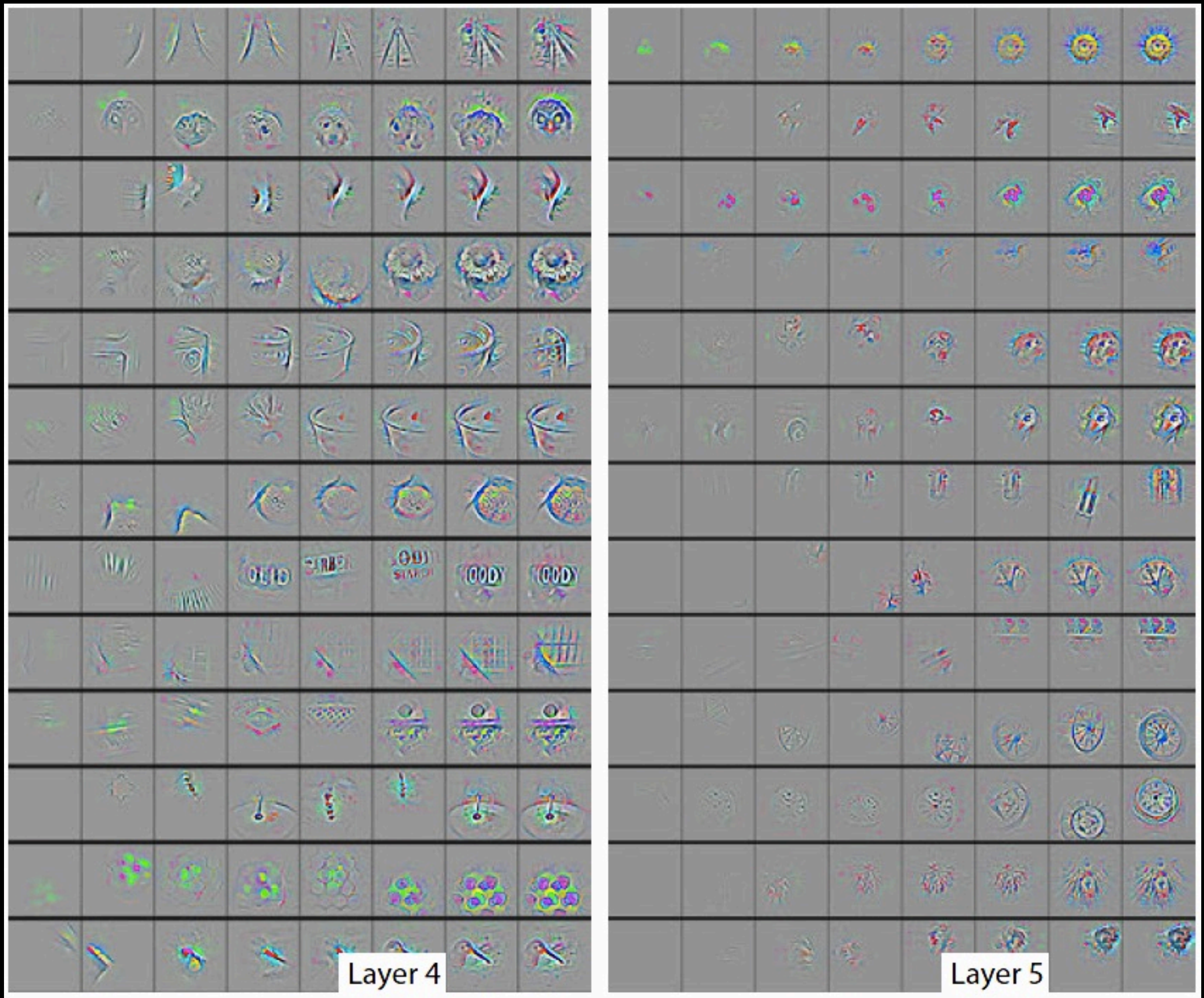
- Start large, slowly reduce
- Explore different scales of energy surface



Evolution of Features During Training



Evolution of Features During Training



Normalization across Data

- Batch Normalization

[Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, Sergey Ioffe, Christian Szegedy, arXiv:1502.03167]

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_\mathcal{B} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_\mathcal{B}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_\mathcal{B})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_\mathcal{B}}{\sqrt{\sigma_\mathcal{B}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

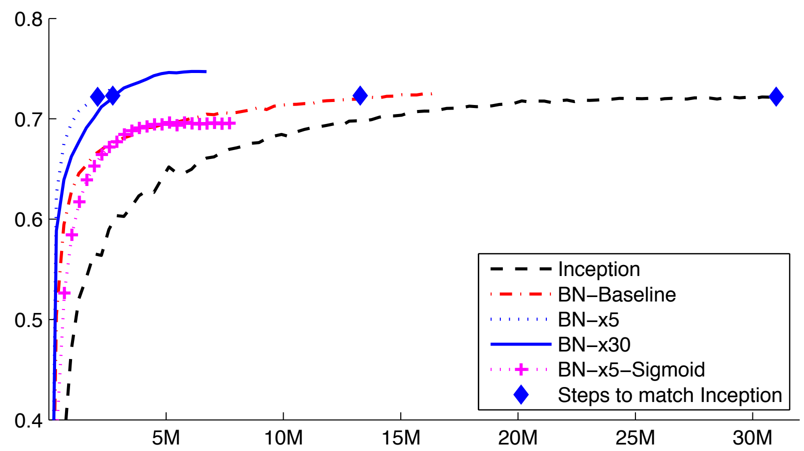


Figure 2: *Single crop validation accuracy of Inception and its batch-normalized variants, vs. the number of training steps.*

Automatic Tuning of Learning Rate?

- ADAGRAD

  J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online leaning and stochastic optimization," in COLT, 2010.

$$\Delta x_t = -\frac{\eta}{\sqrt{\sum_{\tau=1}^{t} g_\tau^2}}\, g_t$$

- ADADELTA

  ADADELTA: An Adaptive Learning Rate Method, Matthew D. Zeiler, arXiv 1212.5701, 2012.

$$\Delta x_t = -\frac{\mathrm{RMS}[\Delta x]_{t-1}}{\mathrm{RMS}[g]_t}\, g_t$$

- No more pesky learning rates

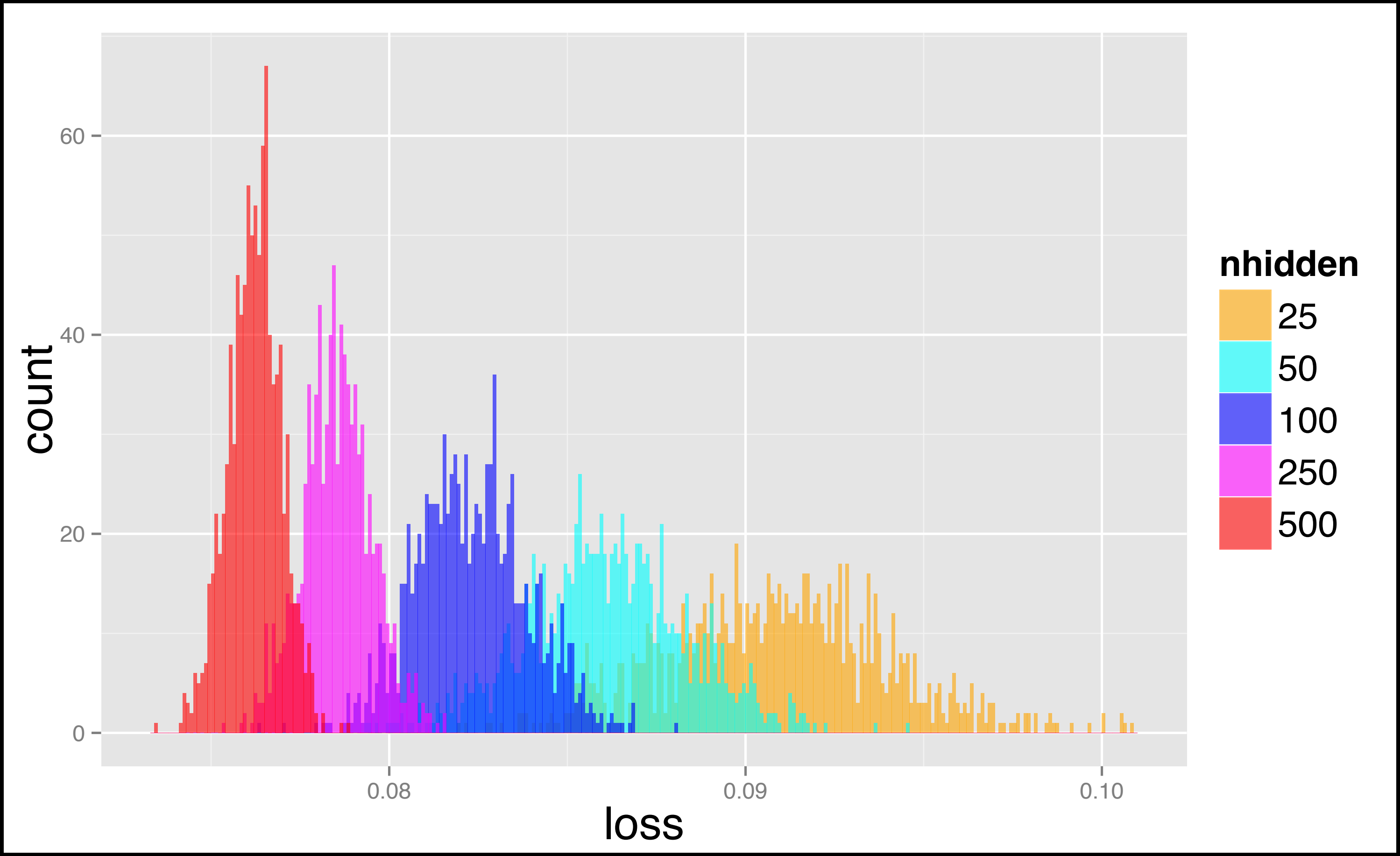
  T. Schaul, S. Zhang, and Y. LeCun, "No more pesky learning rates," arXiv:1206.1106, 2012.

$$\Delta x_t = -\frac{1}{|\mathrm{diag}(H_t)|}\frac{E[g_{t-w:t}]^2}{E[g_{t-w:t}^2]}\, g_t$$

Local Minima?

[The Loss Surfaces of Multilayer Networks
Choromanska et al. http://arxiv.org/pdf/1412.0233v3.pdf]

Distribution of test losses



What about 2nd order methods?

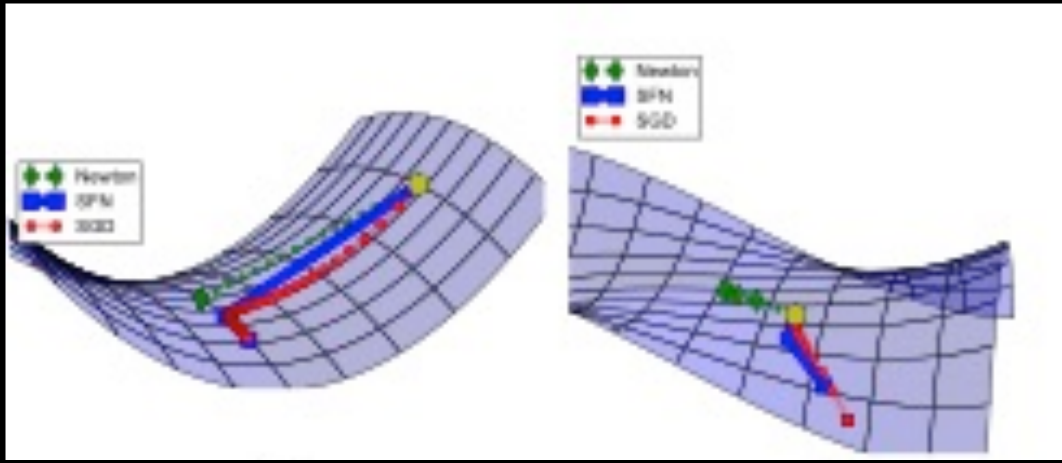
- Newton's method: $\Delta x_t = H_t^{-1} g_t$
- Full Hessian impractical to compute
- Approximations:
  - Diagonal [Becker & Lecun '88] $\Delta x_t = -\frac{1}{|\mathrm{diag}(H_t)| + \mu}\, g_t$
  - Truncated CG [Martens, ICML'10]
  - Per-batch low-rank [Sohl-Dickstien et al., ICML'14]
  - Saddle free (|H|) [Dauphin et al. NIPS'14]

- Generally, extra computation needed seems not worth it: take more (dumb) steps instead!

Saddle Point Perspective

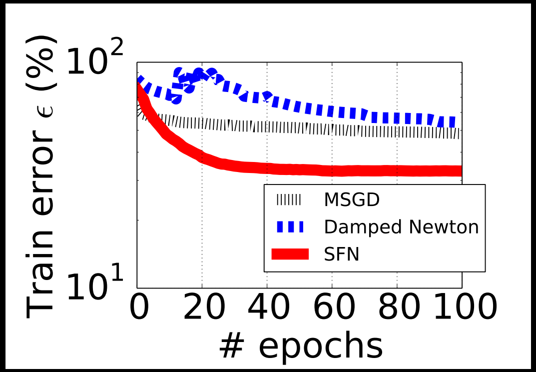
[Identifying and attacking the saddle point problem in high-dimensional non-convex optimization, Dauphin et al., NIPS 2014]

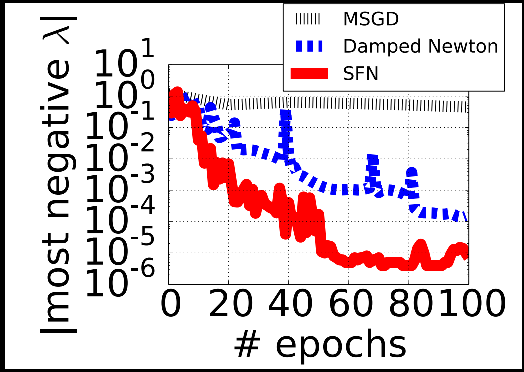
- During optimization Hessian has both +ve and –ve eigenvalues
  - and maybe some zeros too (flat directions)
  - At minimum, all are +ve



- Cause problems for SGD

- Saddle Free Newton (SFN)
  - Use |H| (matrix where take absolute value of each eigenvalue of H)



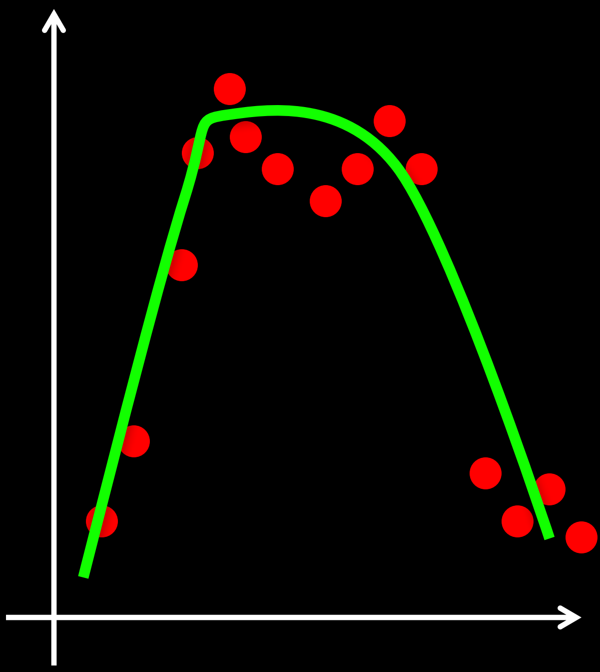


Improving Generalization

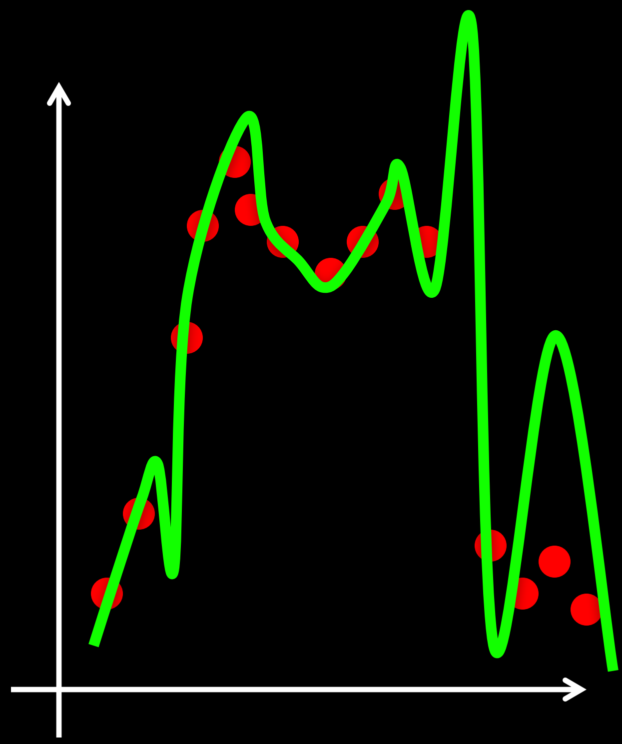
- Data Augmentation (jitter, peturb)
- Weight decay (L1/2 penalty on weights)
- Weight sharing (reduces # parameters)
- Multi-task learning
- Inject Noise into network
  - DropOut [Hinton et al. 2012]
  - DropConnect [Wan et al. ICML 2012]
  - Stochastic Pooling [Zeiler & Fergus ICLR'13]

Big Model + Regularize vs Small Model

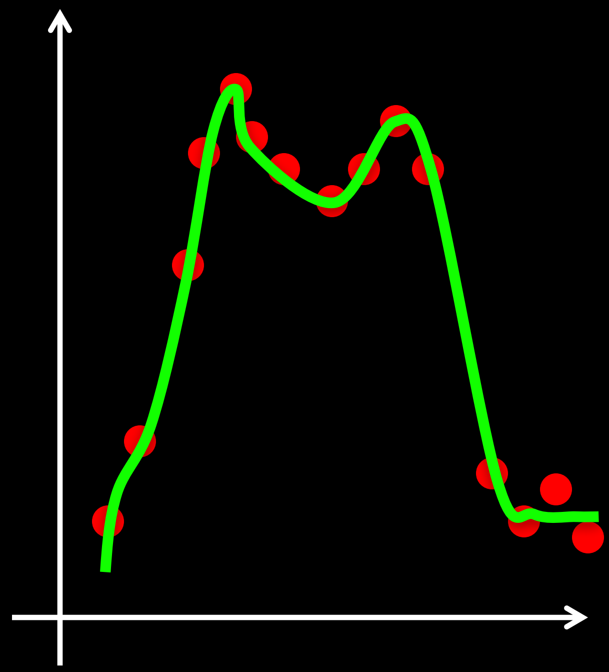
Small model



Big model



Big model
+ Regularize



Fooling Convnets

- Search for images that are misclassified by the network
- Intriguing properties of neural networks, Christian Szegedy et al. arXiv 1312.6199, 2013
- Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images, Anh Nguyen, Jason Yosinski, Jeff Clune, arXiv 1412.1897.
- Problem common to any discriminative method

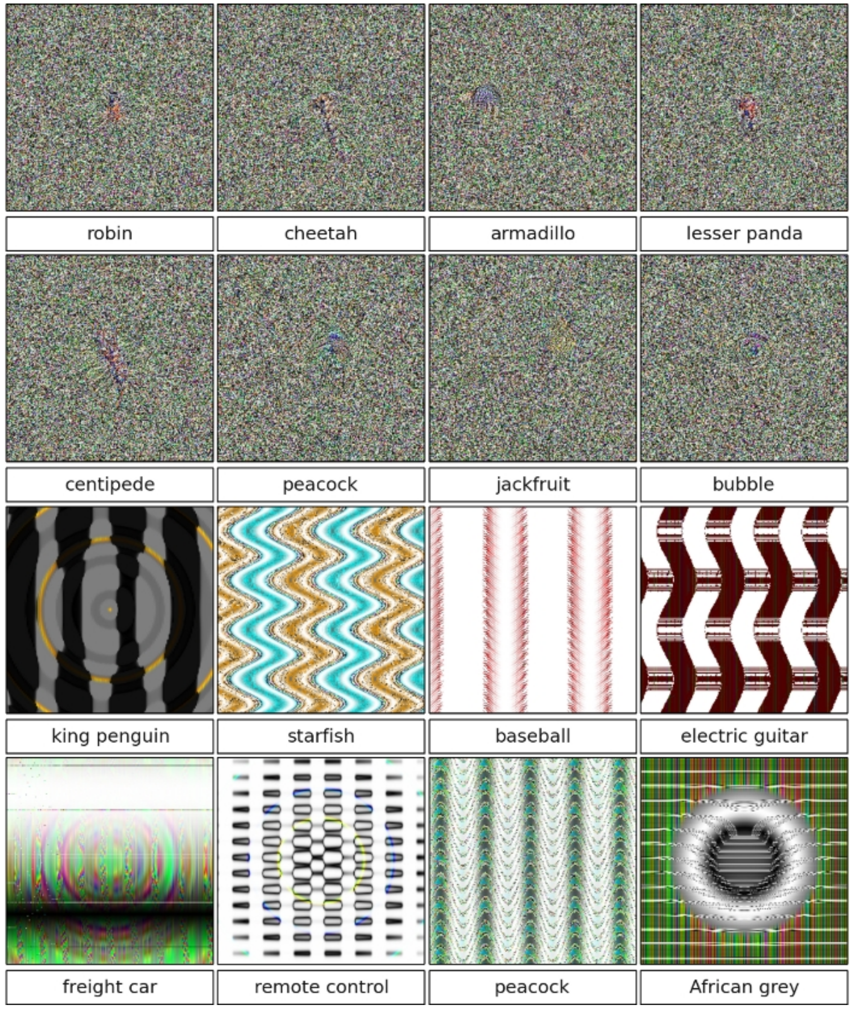
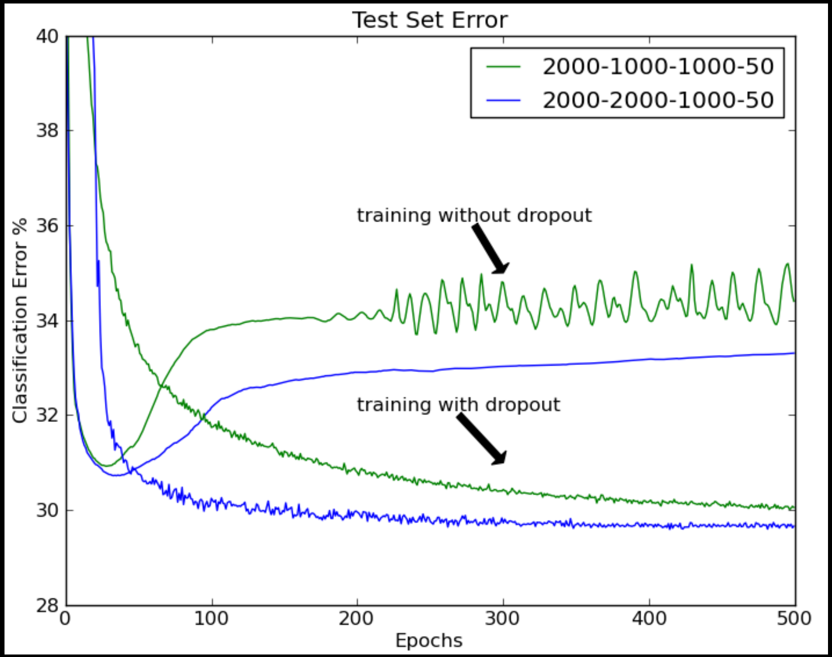


Figure 1. Evolved images that are unrecognizable to humans, but that state-of-the-art DNNs trained on ImageNet believe with $\geq$ 99.6% certainty to be a familiar object. This result highlights differences between how DNNs and humans recognize objects.

DropOut

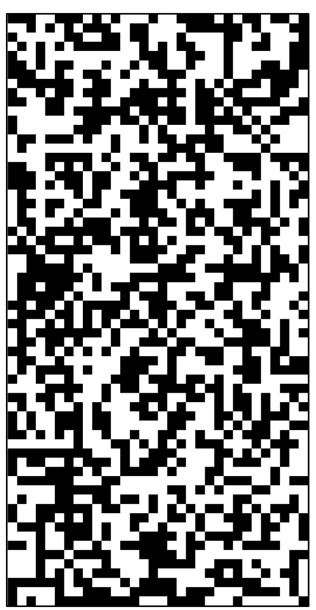
- G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever and R. R. Salakhutdinov, *Improving neural networks by preventing co-adaptation of feature detectors,* arXiv:1207.0580 2012

- Fully connected layers only
- Randomly set activations in layer to zero
- Gives ensemble of models
- Similar to bagging [Breiman'94], but differs in that parameters are shared.

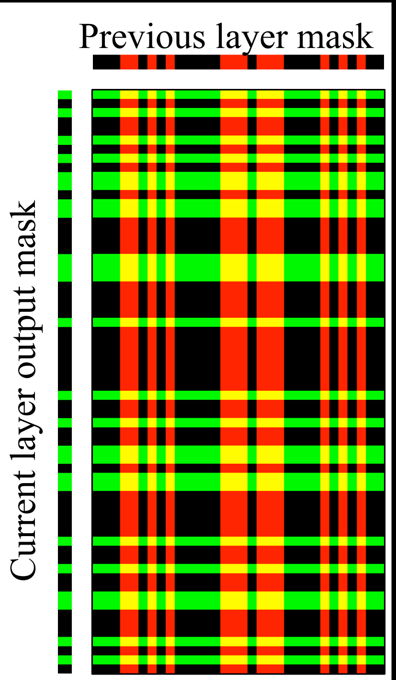


DropConnect

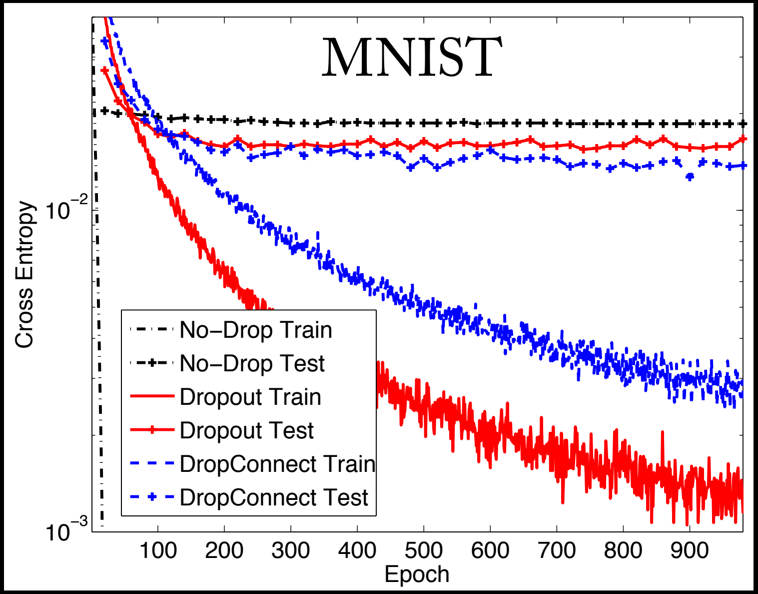
- Wan et al. ICML 2013
- Fully-connected layers only
- Random binary mask on weights



b) DropConnect mask M



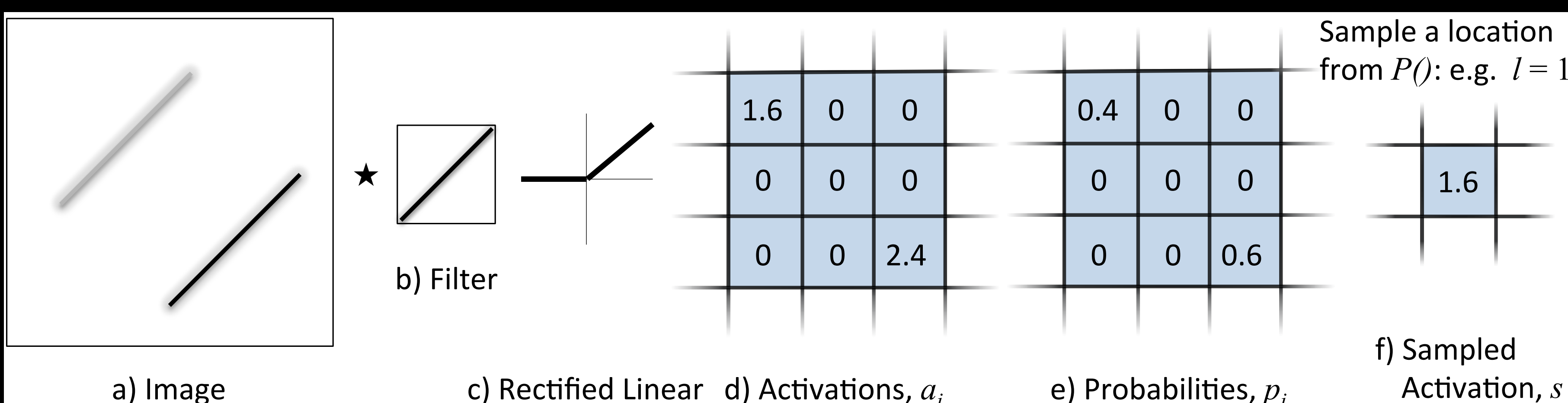
c) Effective Dropout mask M'



Stochastic Pooling

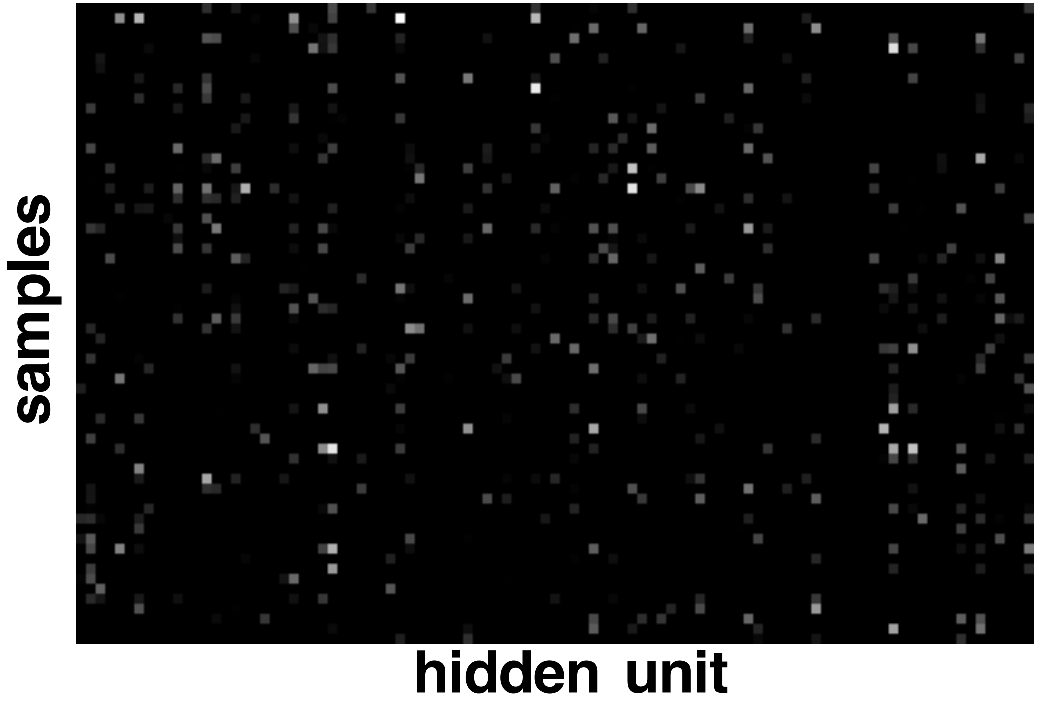
[Zeiler and Fergus, ICLR 2013]

- For conv layers
- Compute activations a_i: (≥ 0)
- Normalize to sum to 1 -> $p_i = \frac{a_i}{\sum_{k \in R_j} a_k}$
- Sample location, l, from multinomial
- Use activation from the location: $s = a_l$



OTHER THINGS GOOD TO KNOW

- Check gradients numerically by finite differences
- Visualize features (feature maps need to be uncorrelated) and have high variance.

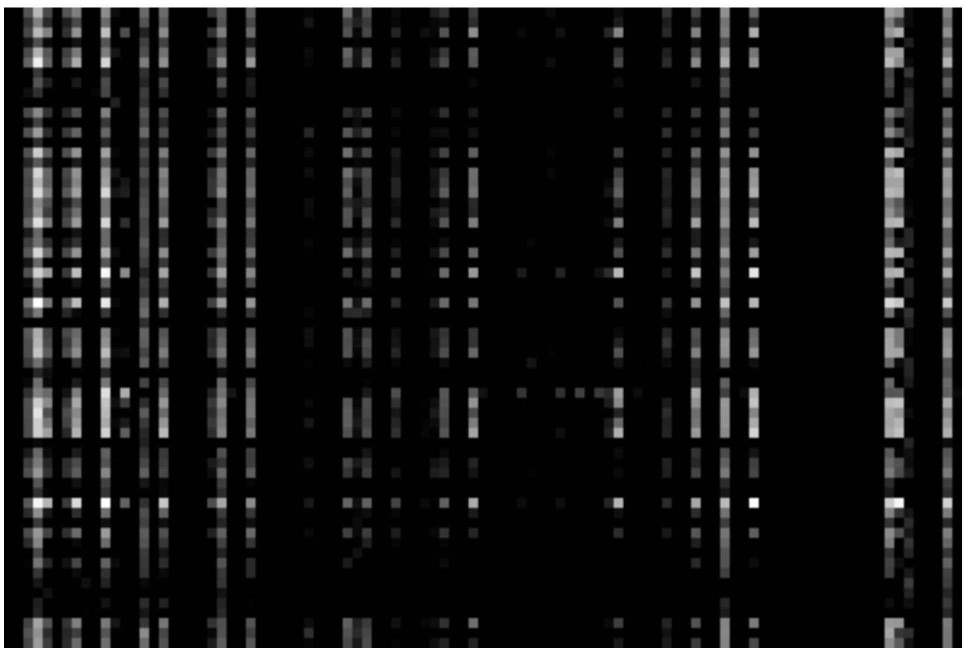


Good training: hidden units are sparse across samples and across features.

Ranzato

OTHER THINGS GOOD TO KNOW

- Check gradients numerically by finite differences
- Visualize features (feature maps need to be uncorrelated) and have high variance.

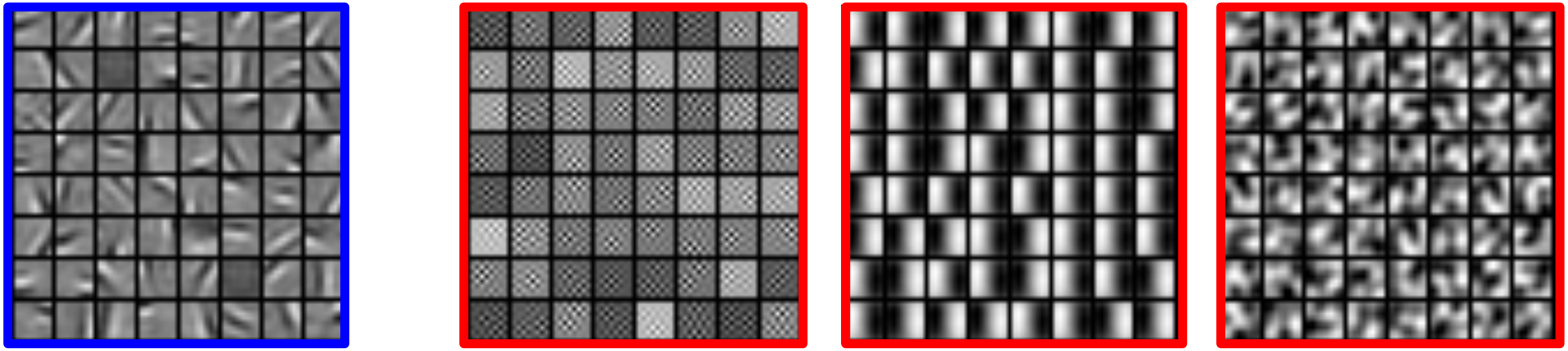


Bad training: many hidden units ignore the input and/or exhibit strong correlations.

OTHER THINGS GOOD TO KNOW

- Check gradients numerically by finite differences
- Visualize features (feature maps need to be uncorrelated) and have high variance.
- Visualize parameters

GOOD BAD BAD BAD



too noisy too correlated lack structure

Good training: learned filters exhibit structure and are uncorrelated.

OTHER THINGS GOOD TO KNOW

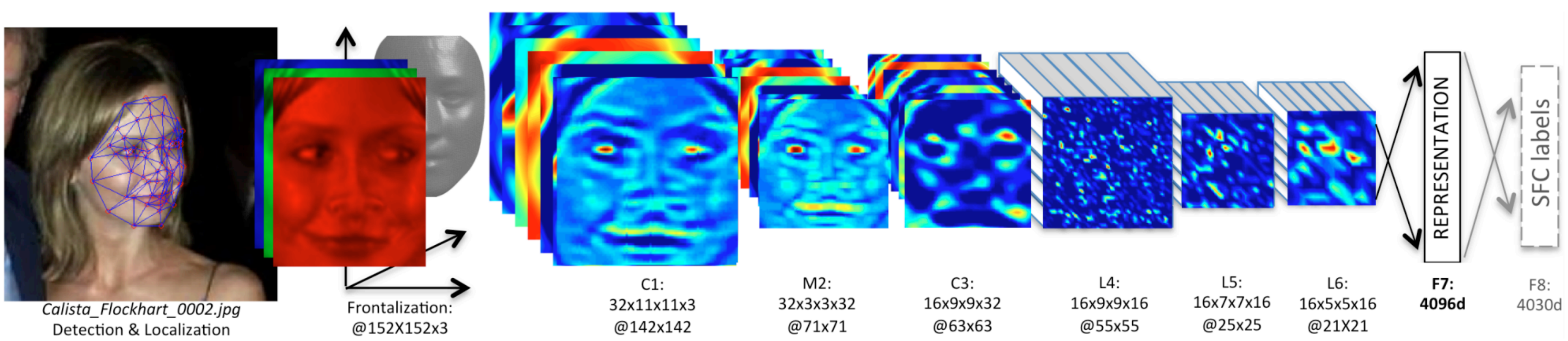
- Check gradients numerically by finite differences
- Visualize features (feature maps need to be uncorrelated) and have high variance.
- Visualize parameters
- Measure error on both training and validation set.
- Test on a small subset of the data and check the error → 0.

WHAT IF IT DOES NOT WORK?

- Training diverges:
  - Learning rate may be too large → decrease learning rate
  - BPROP is buggy → numerical gradient checking
- Parameters collapse / loss is minimized but accuracy is low
  - Check loss function:
    - Is it appropriate for the task you want to solve?
    - Does it have degenerate solutions? Check "pull-up" term.
- Network is underperforming
  - Compute flops and nr. params. → if too small, make net larger
  - Visualize hidden units/params → fix optmization
- Network is too slow
  - Compute flops and nr. params. → GPU,distrib. framework, make net smaller

Industry Deployment

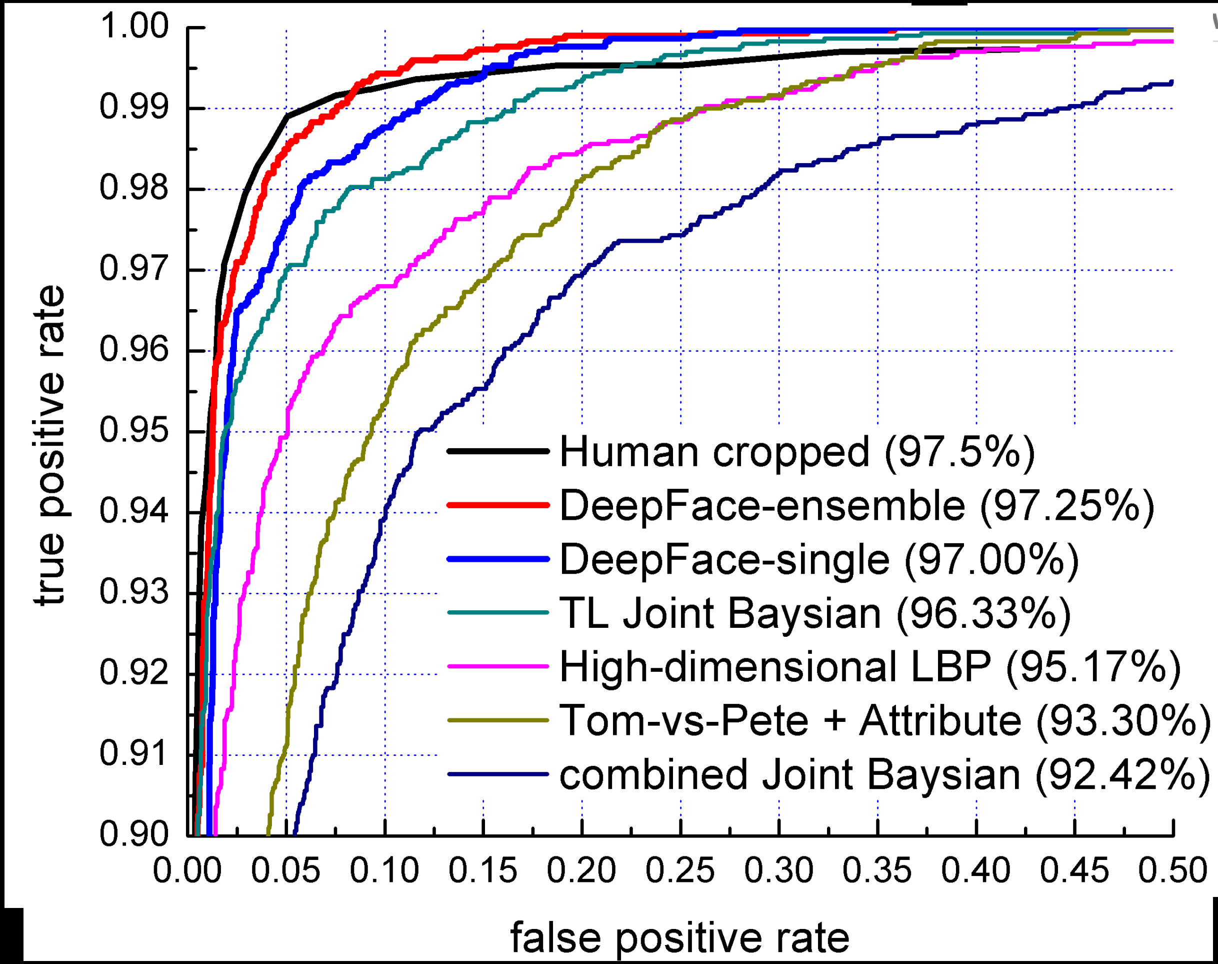
- Used in Facebook, Google, Microsoft
- Face recognition, image search, photo organization….
- Very fast at test time (~100 images/sec/GPU)

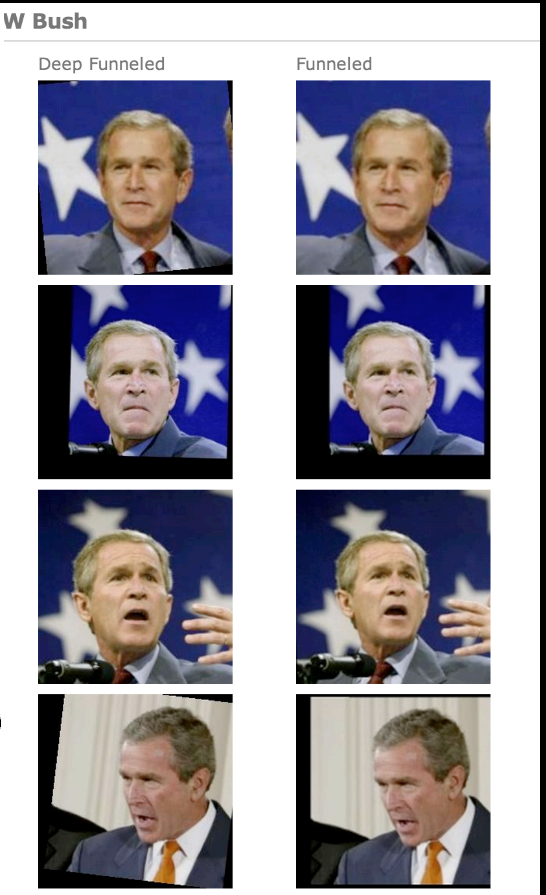


[Taigman et al. DeepFace: Closing the Gap to Human-Level Performance in Face Verification, CVPR'14]

Labeled Faces in Wild Dataset

- Task: given pair of images, same person or not?

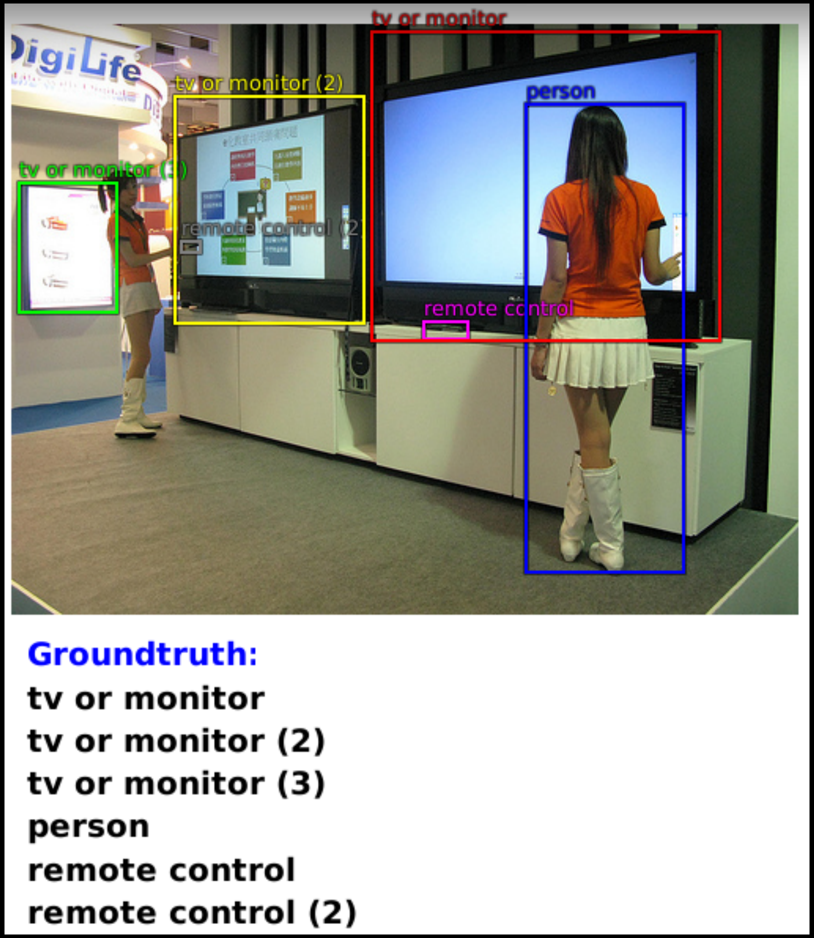




[Tagman et al. CVPR'14]

Detection with ConvNets

- So far, all about classification
- What about localizing objects within the scene?

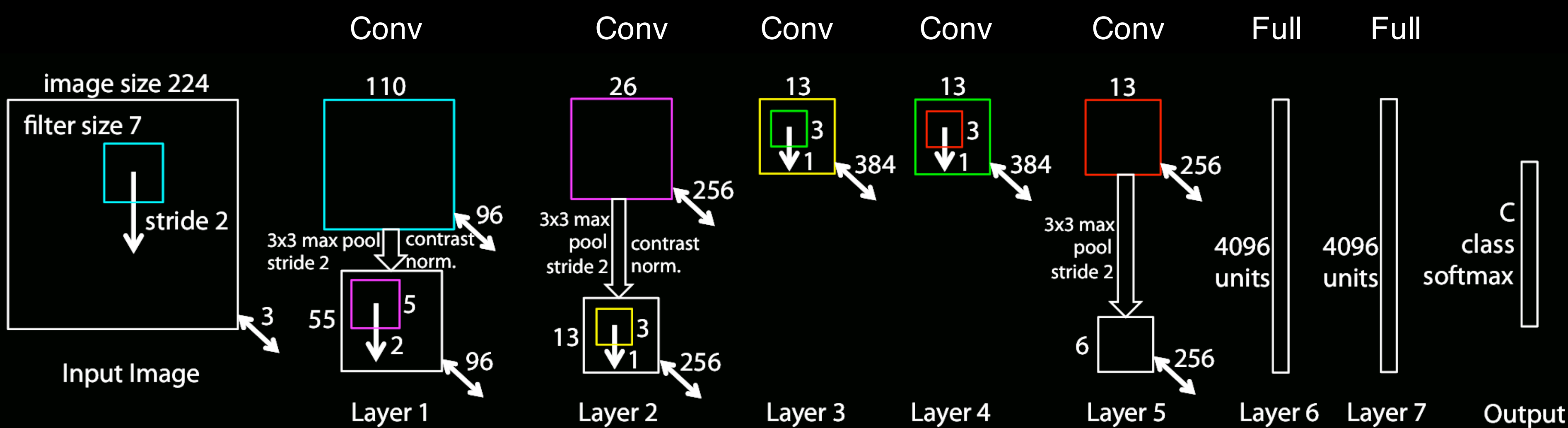


Two General Approaches

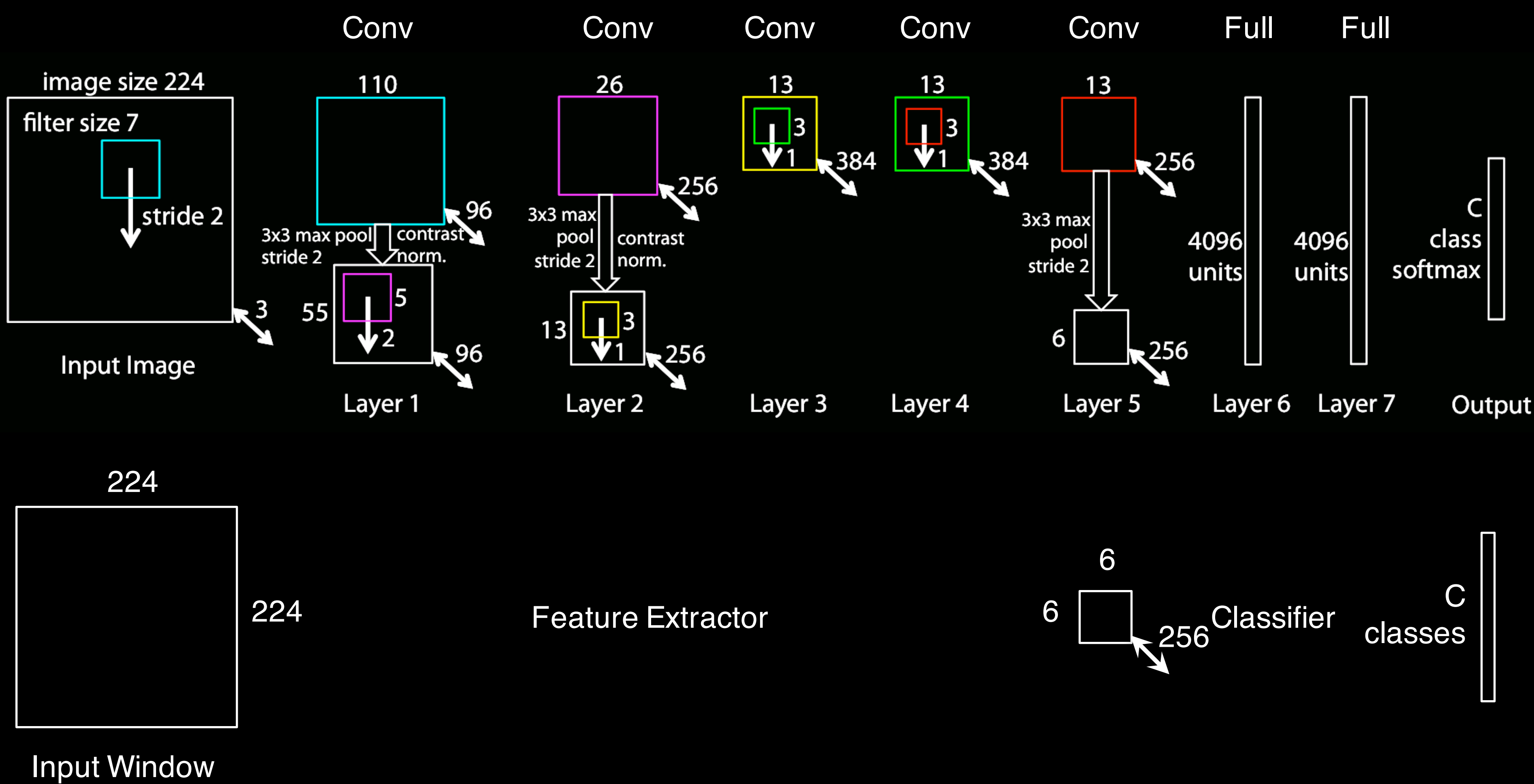
1. Examine very position / scale
   - E.g. Overfeat: Integrated recognition, localization and detection using convolutional networks, Sermanet et al., ICLR 2014

2. Use some kind of proposal mechanism to attend to a set of possible regions
   - E.g. Region-CNN [Rich feature hierarchies for accurate object detection and semantic segmentation, Girshick et al., CVPR 2014]

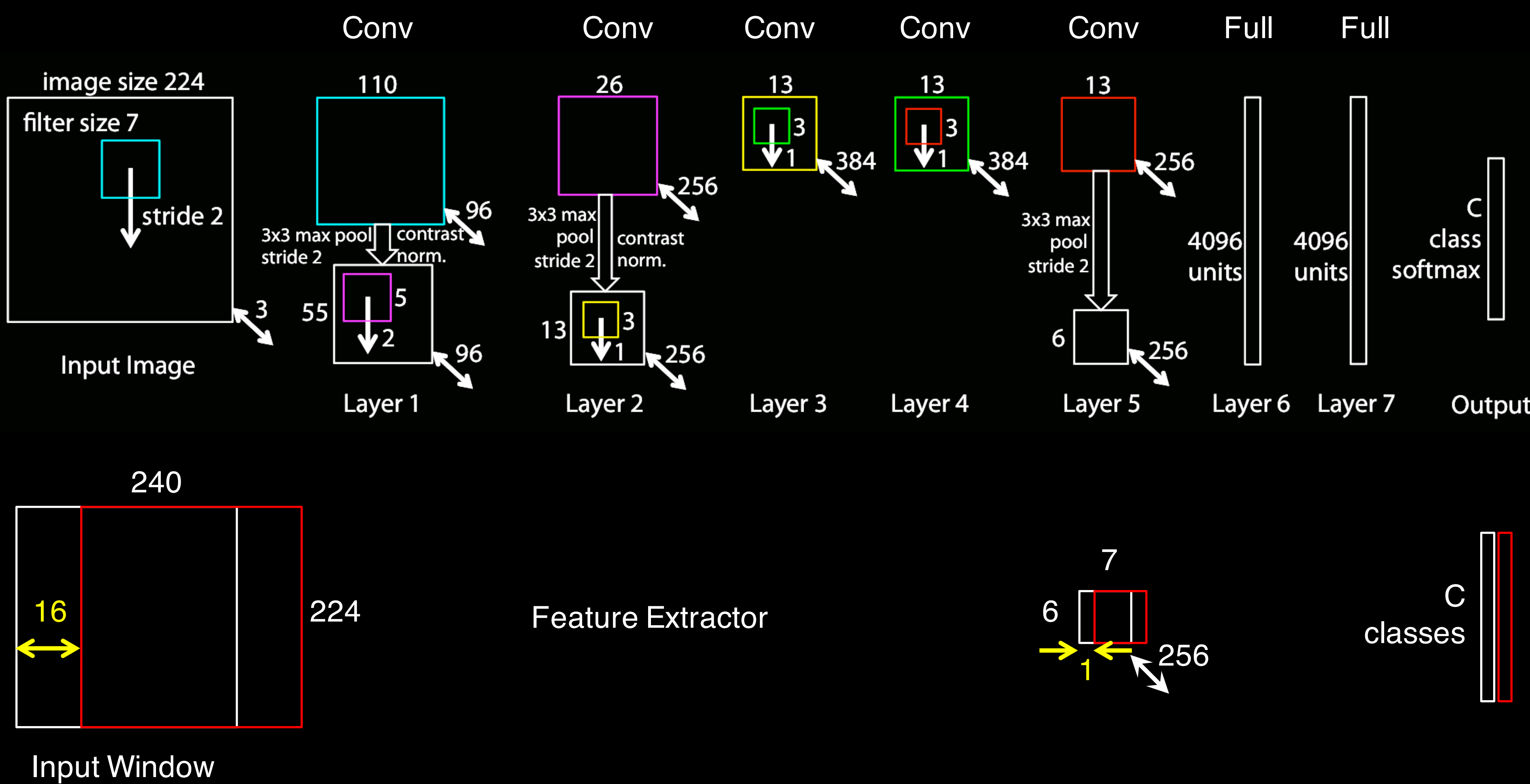
Sliding Window with ConvNet



Sliding Window with ConvNet

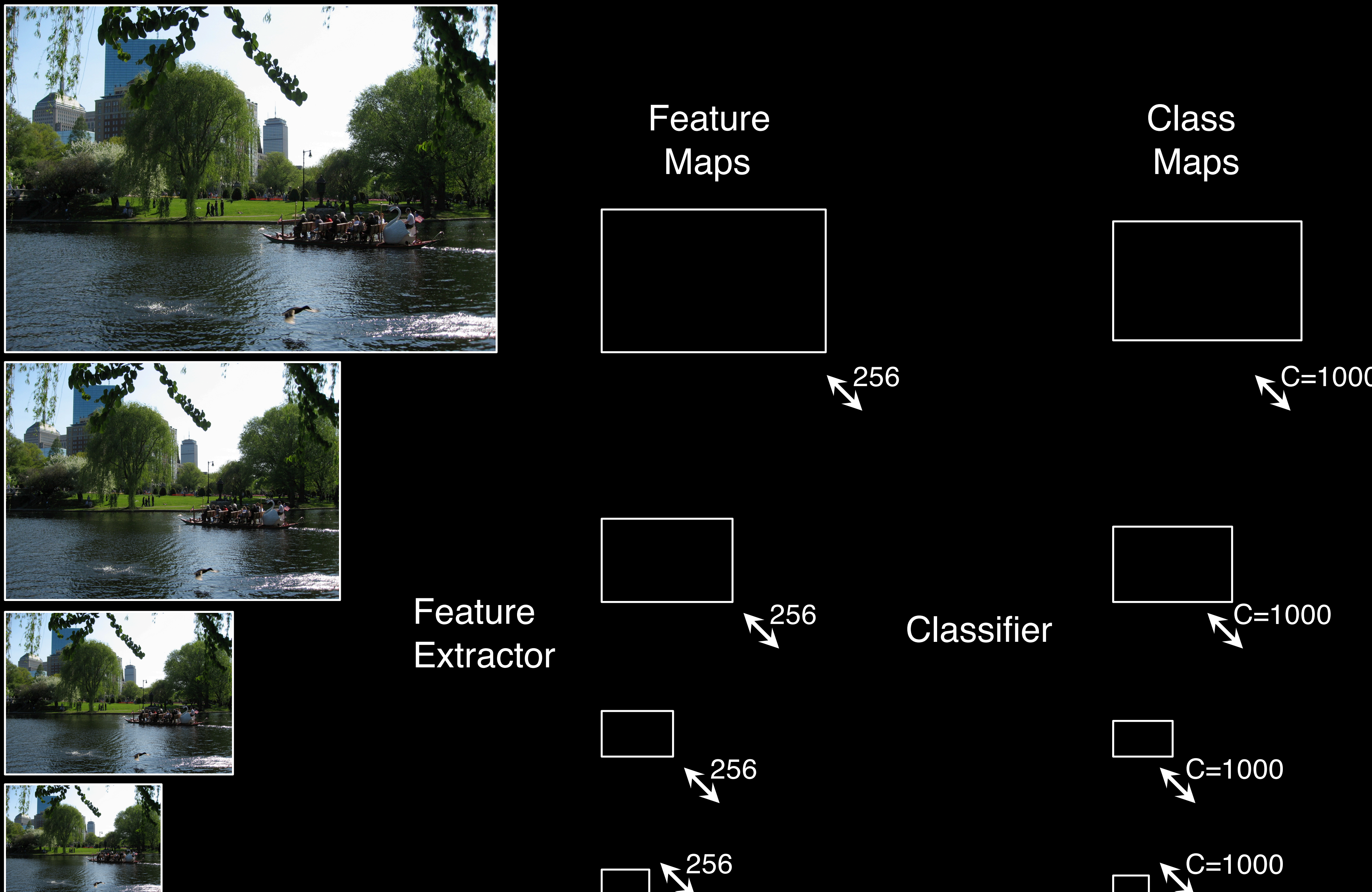


Sliding Window with ConvNet

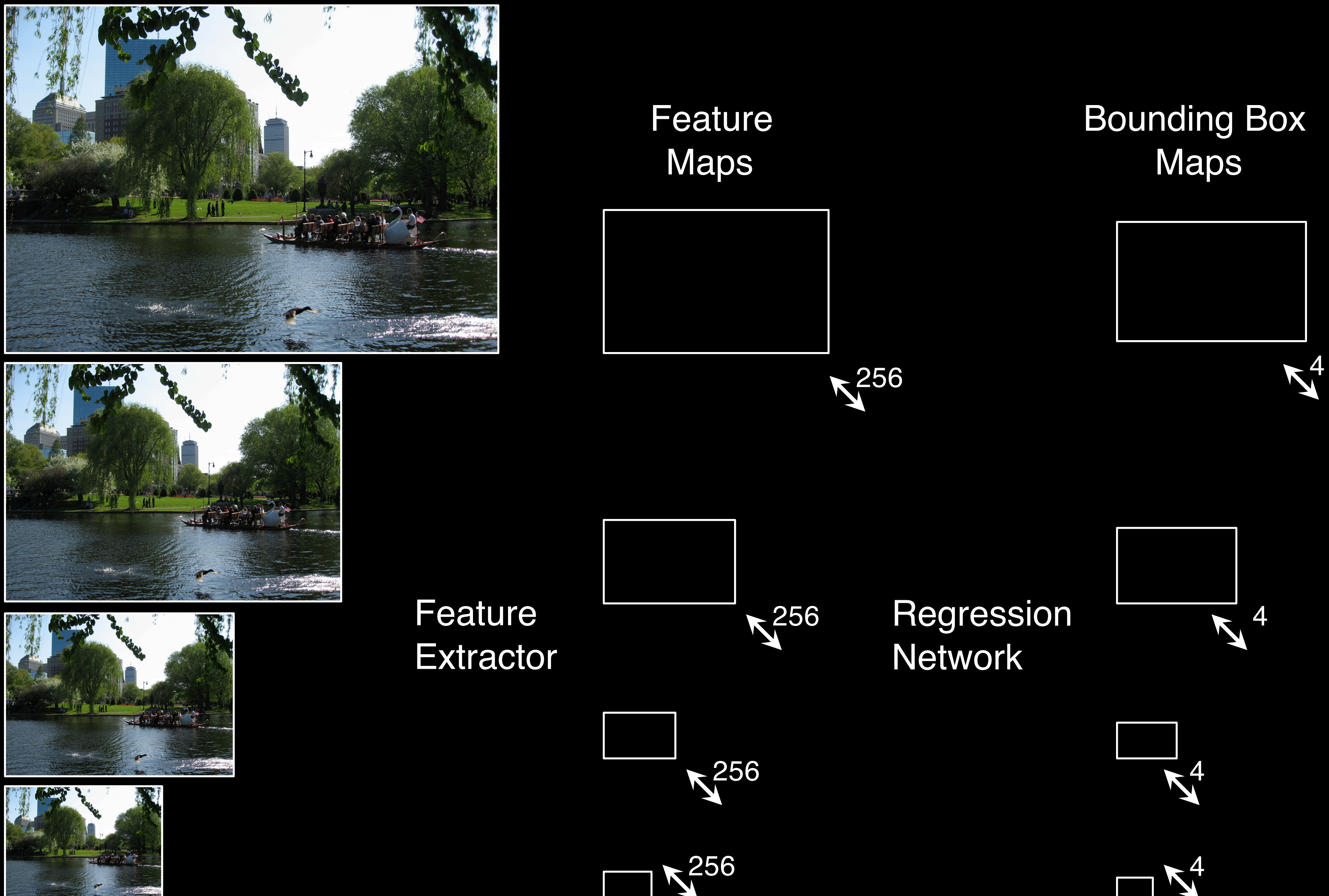


No need to compute two separate windows --- Just one big input window

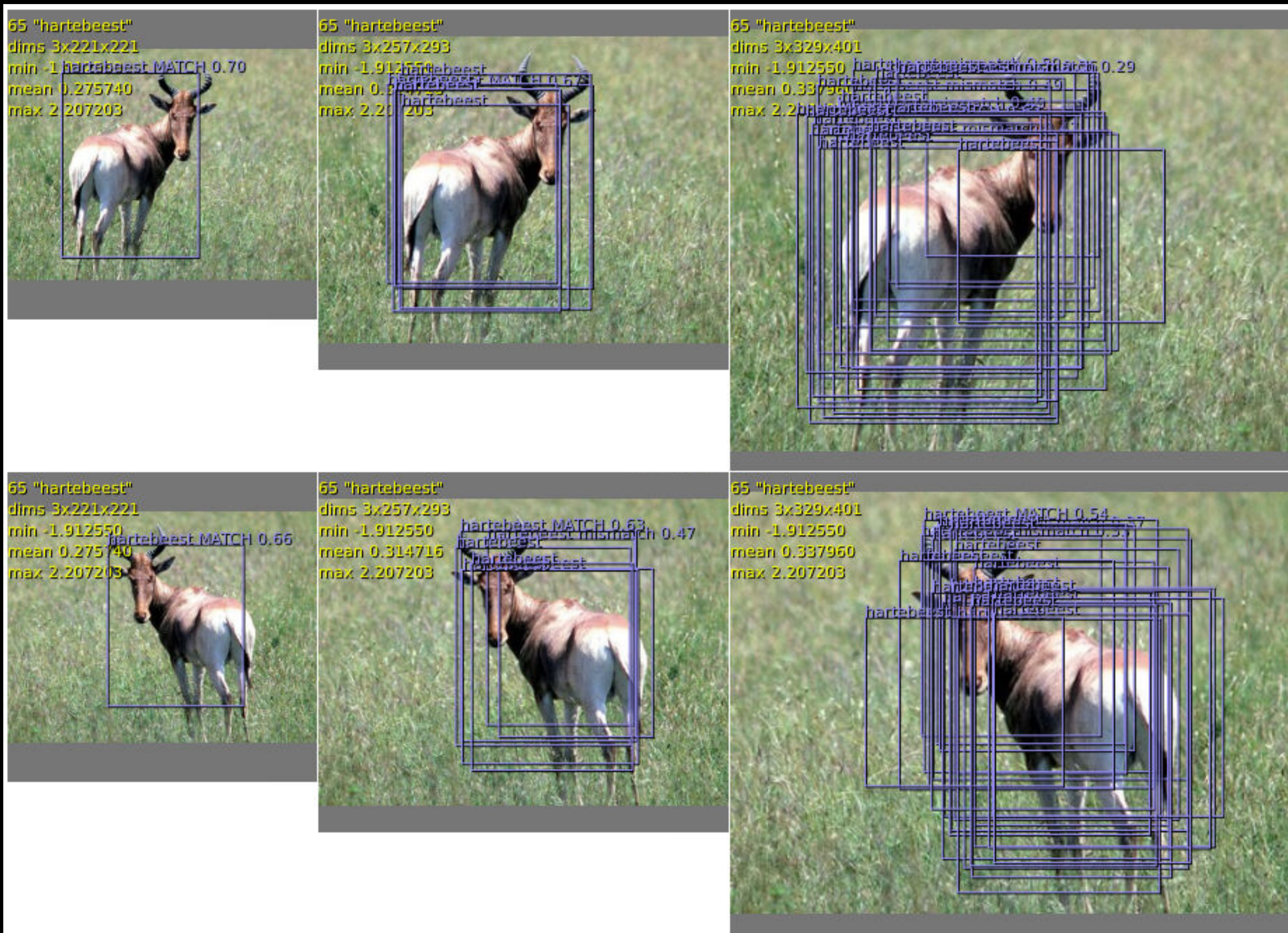
Multi-Scale Sliding Window ConvNet



Multi-Scale Sliding Window ConvNet

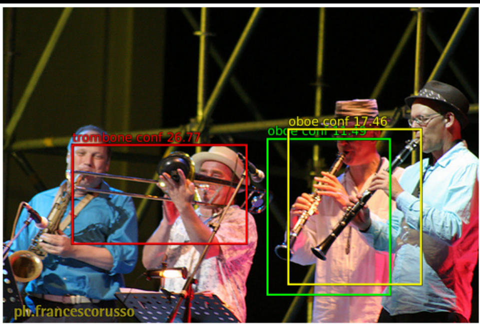


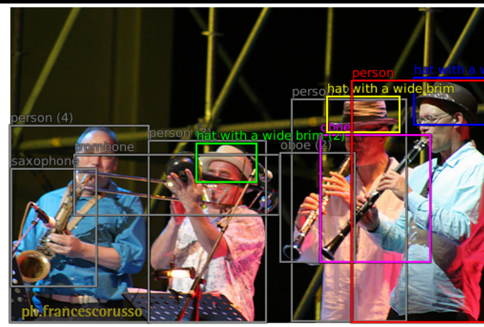
OverFeat – Output before NMS



Overfeat Detection Results

[Sermanet et al. ICLR 2014]

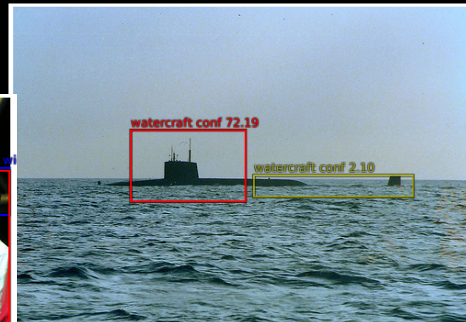


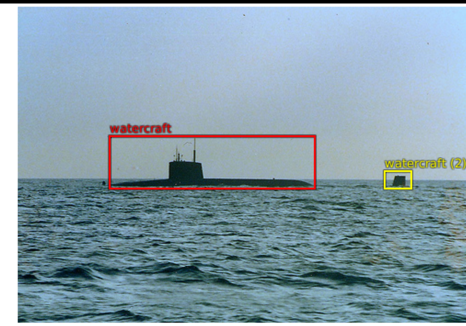


Top predictions:
trombone (confidence 26.8)
oboe (confidence 17.5)
oboe (confidence 11.5)

ILSVRC2012_val_00000614.JPEG

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

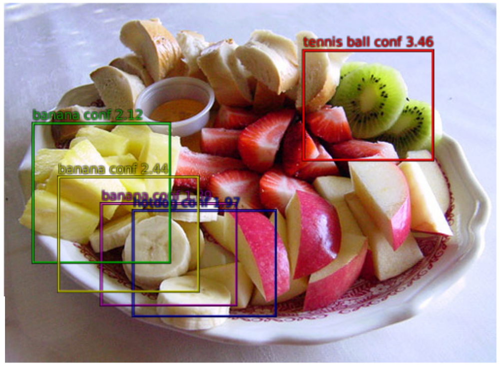


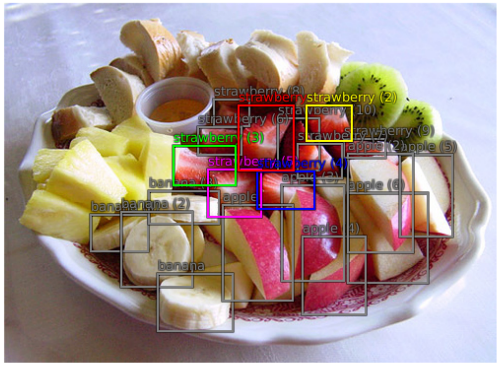


Top predictions:
watercraft (confidence 72.2)
watercraft (confidence 2.1)

ILSVRC2012_val_00000623.JPEG

Groundtruth:
watercraft
watercraft (2)





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

ILSVRC2012_val_00000320.JPEG

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

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

ILSVRC2012_val_00000519.JPEG

Groundtruth:
bowl
microwave

R-CNN Approach

[Rich feature hierarchies for accurate object detection and semantic segmentation, Girshick et al., CVPR 2014]

- Bottom-up proposa
mechanism
- Scored by classifier

- Current best detection approach on PASCAL VOC

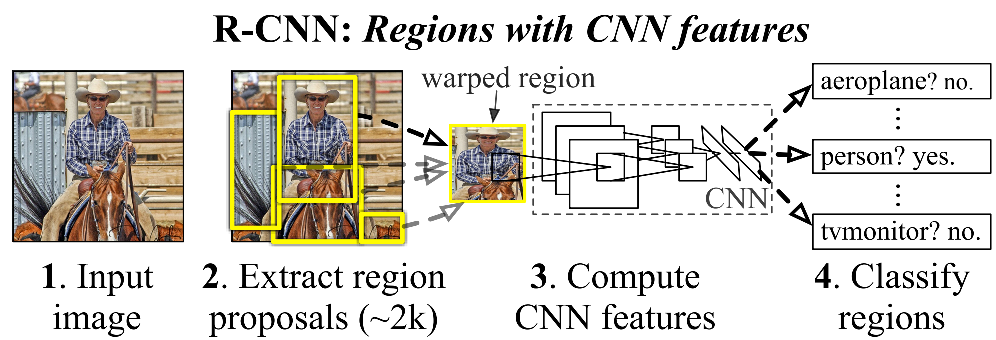
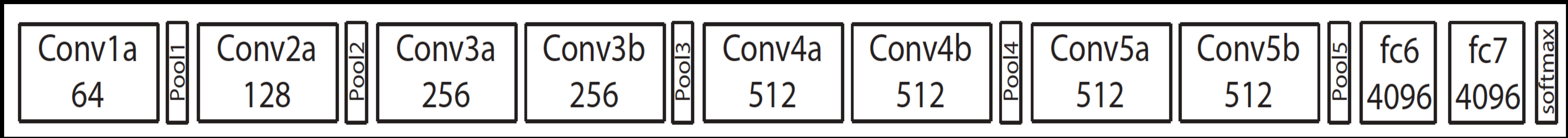


Figure 1: Object detection system overview. Our system (1) takes an input image, (2) extracts around 2000 bottom-up region proposals, (3) computes features for each proposal using a large convolutional neural network (CNN), and then (4) classifies each region using class-specific linear SVMs. R-CNN achieves a mean average precision (mAP) of **53.7% on PASCAL VOC 2010**. For comparison, [34] reports 35.1% mAP using the same region proposals, but with a spatial pyramid and bag-of-visual-words approach. The popular deformable part models perform at 33.4%.

- Further work combines proposal mechanism with classification network:
  - Fast R-CNN, Ross Girshick, arXiv 1504.08083, 2015.
  - Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, Shaoqing Ren et al., arXiv 1506.01497, 2015

Video Classification

- Want to capture temporal structure
- 3D convolutions & 3D max-pooling
- E.g. C3D model



8 convolution, 5 pool, 2 fully-connected layers
3x3x3 convolution kernels
2x2x2 pooling kernels

[Learning Spatiotemporal Features with 3D Convolutional Networks, Tran et al., arXiv:1412.0767, 2014]

[Slide: Manohar Paluri]

Action Recognition – UCF101 dataset

[Slide: Manohar Paluri]

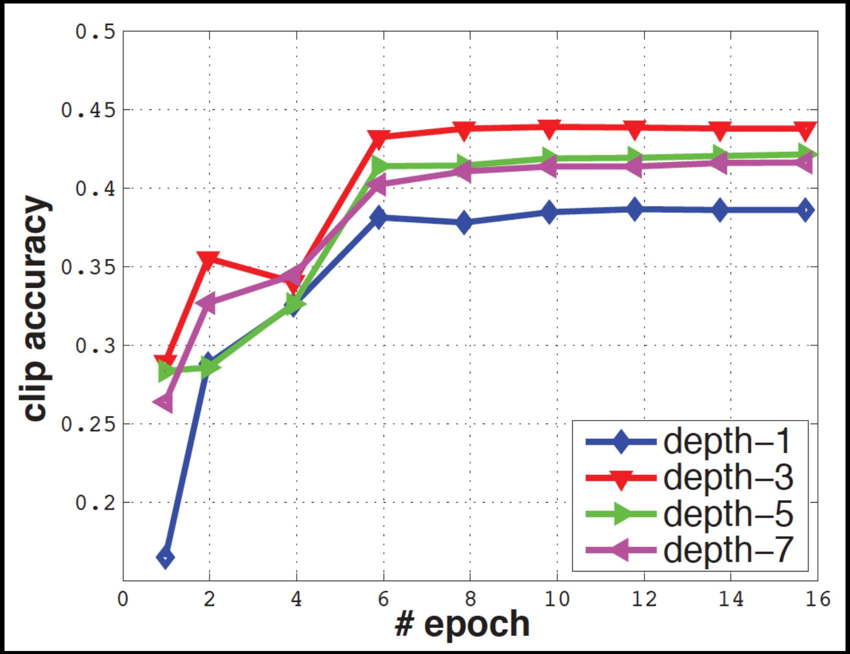
Action Recognition Results

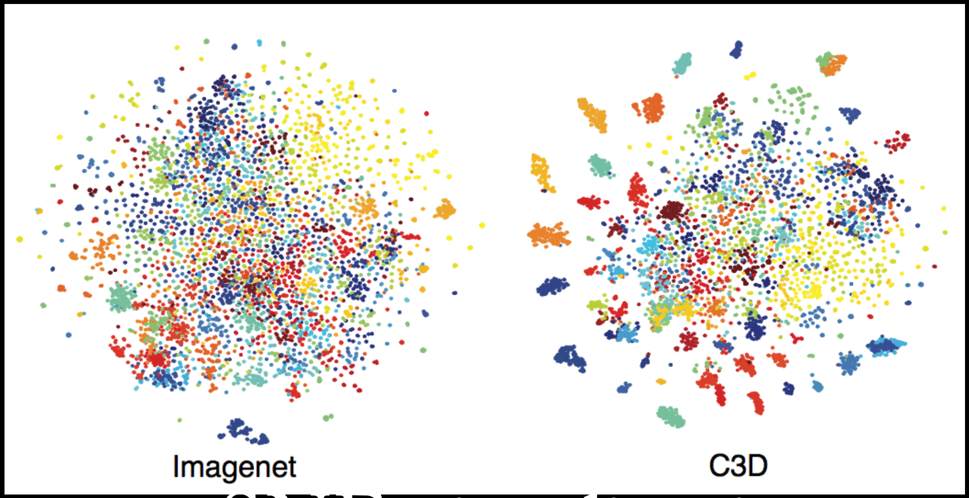
	Method	Accuracy (%)
Baselines	Imagenet	68.8
	iDT	76.2
Use raw pixel inputs	Deep networks [19]	65.4
	Spatial stream network [36]	72.6
	LRCN [7]	71.1
	LSTM composite model [39]	75.8
	C3D (1 net)	82.3
	C3D (3 nets)	**85.2**
Use optical flows	iDT with Fisher vector [31]	87.9
	Temporal stream network [36]	83.7
	Two-stream networks [36]	88.0
	LRCN [7]	82.9
	LSTM composite model [39]	84.3
	Multi-skip feature stacking [26]	89.1
	C3D (3 nets) + iDT	**90.4**

[Slide: Manohar Paluri]

2D vs 3D Convnets

- UCF101 training





t-SNE visualization

[Slide: Manohar Paluri]

Sport Classification Results

Method	Number of Nets	Clip hit@1	Video hit@1	Video hit@5
Deep Video's Single-Frame + Multires [19]	3 nets	42.4	60.0	78.5
Deep Video's Slow Fusion [19]	1 net	41.9	60.9	80.2
C3D (trained from scratch)	1 net	44.9	60.0	84.4
C3D (fine-tuned from I380K pre-trained model)	1 net	**46.1**	**61.1**	**85.2**

[Slide: Manohar Paluri]

Dense Scene Labeling

- Classification: pixels -> label
- Detection: pixels -> boxes

- Use Convnets to do pixels -> pixels
  - Segmentation of image
  - Image processing tasks (denoising etc.)
  - Don't want pooling

Dense Scene Labeling

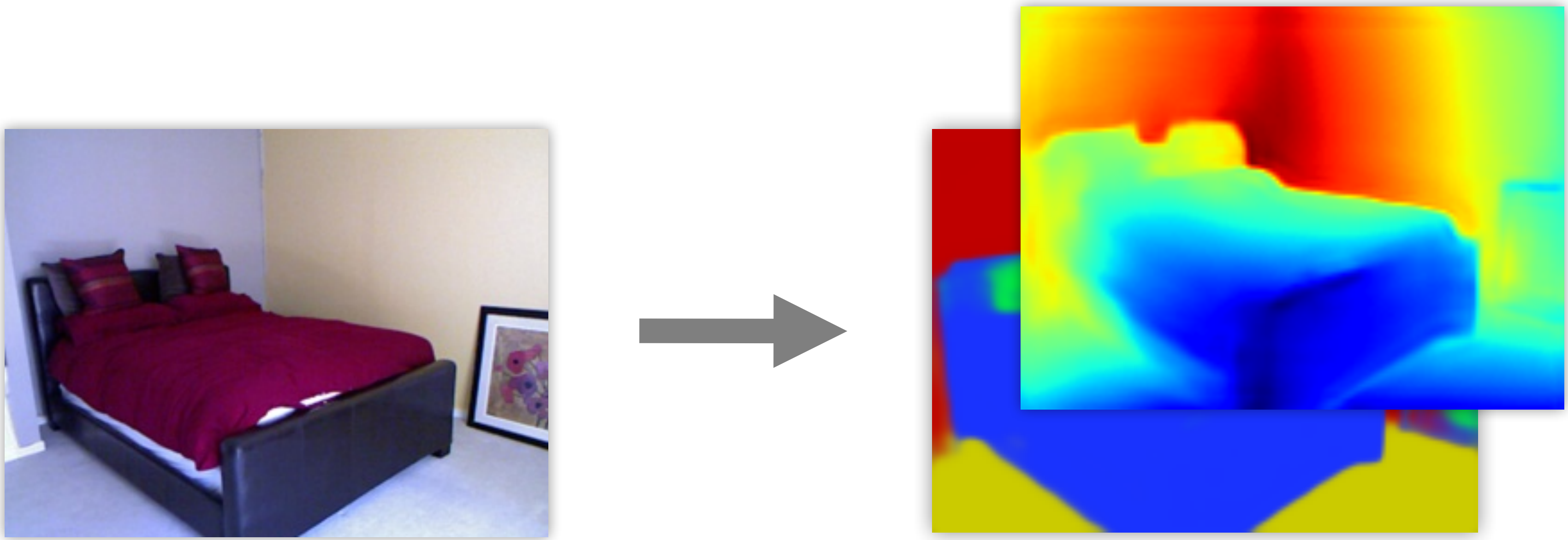
Input Image

Semantic Map

- Convnet output is per-pixel label map

Dense Scene Labeling

Depth

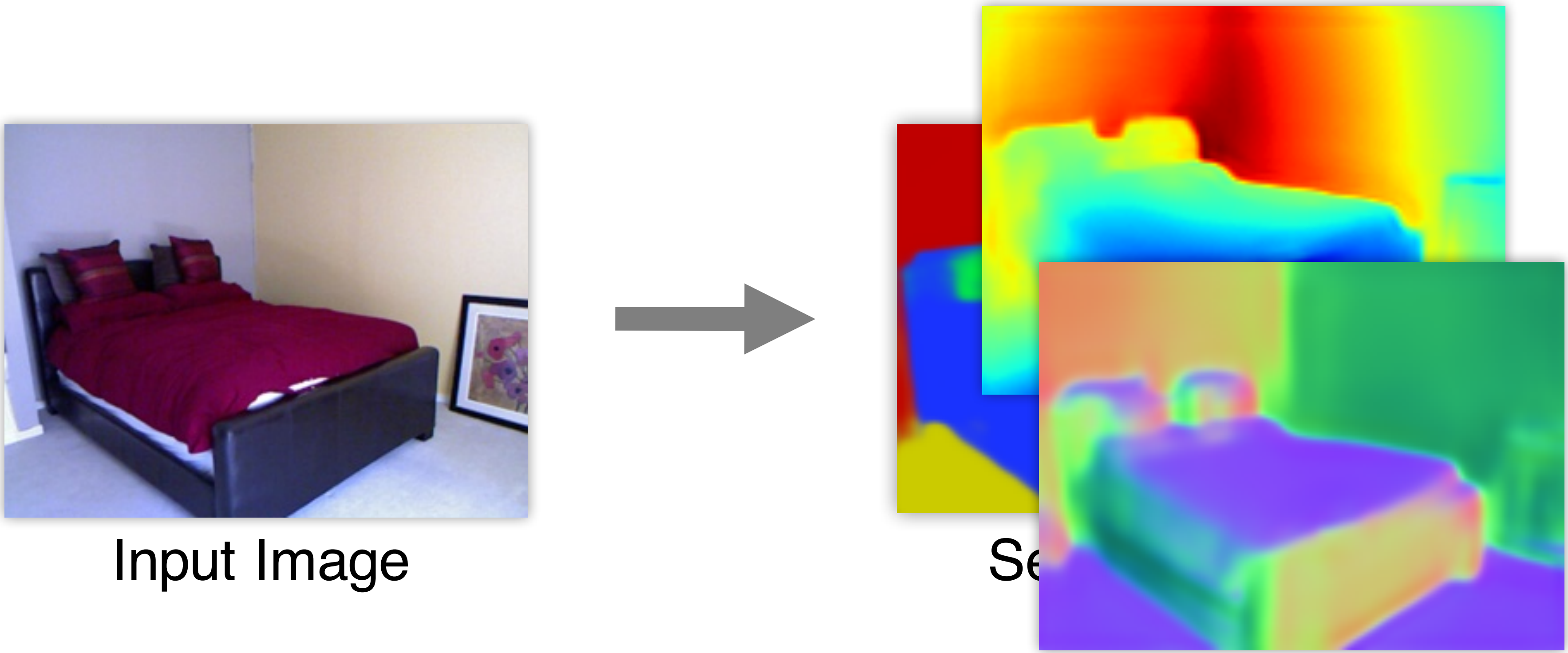


Input Image

Semantic Map

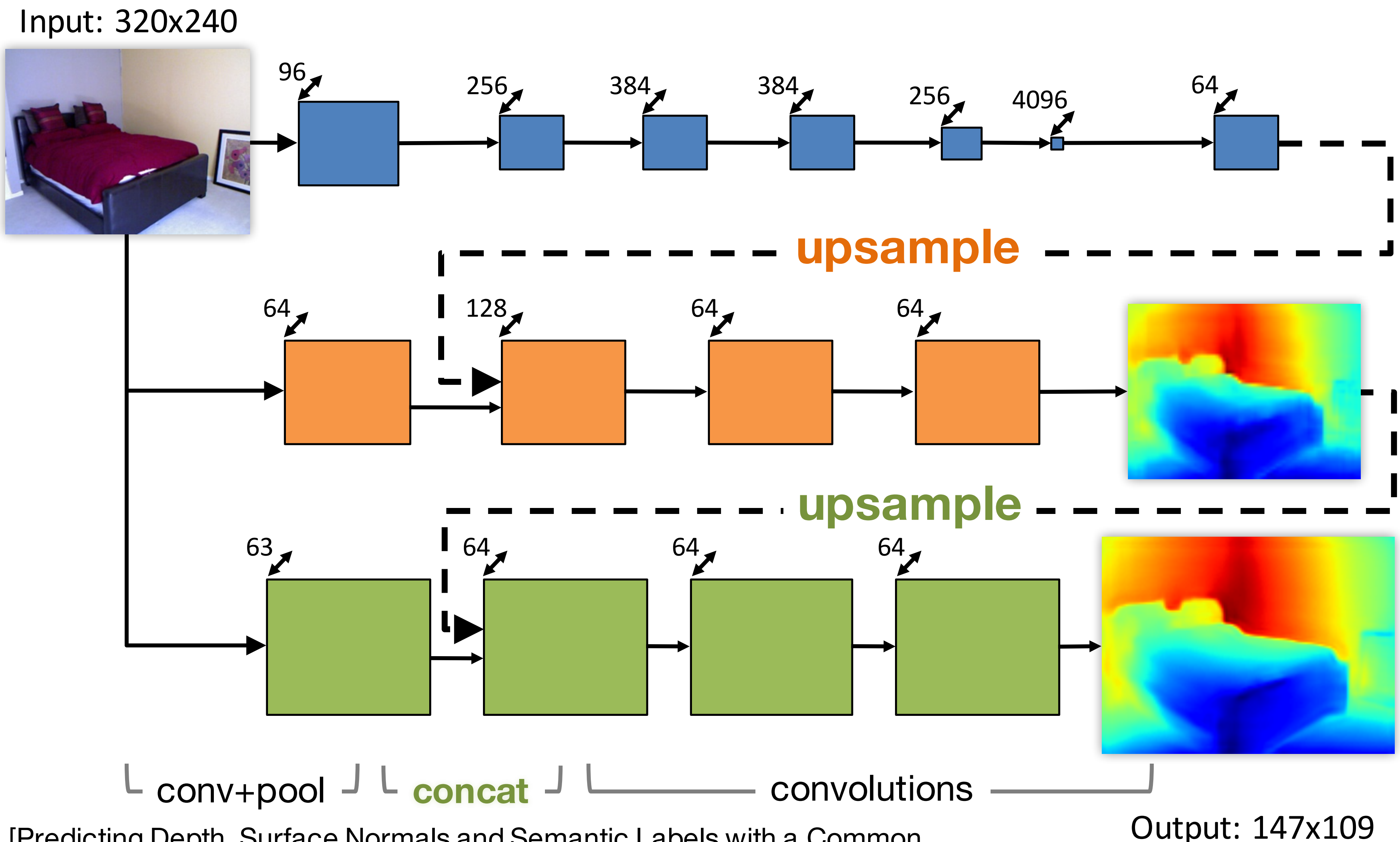
- Convnet output is per-pixel depth map

Dense Scene Labeling



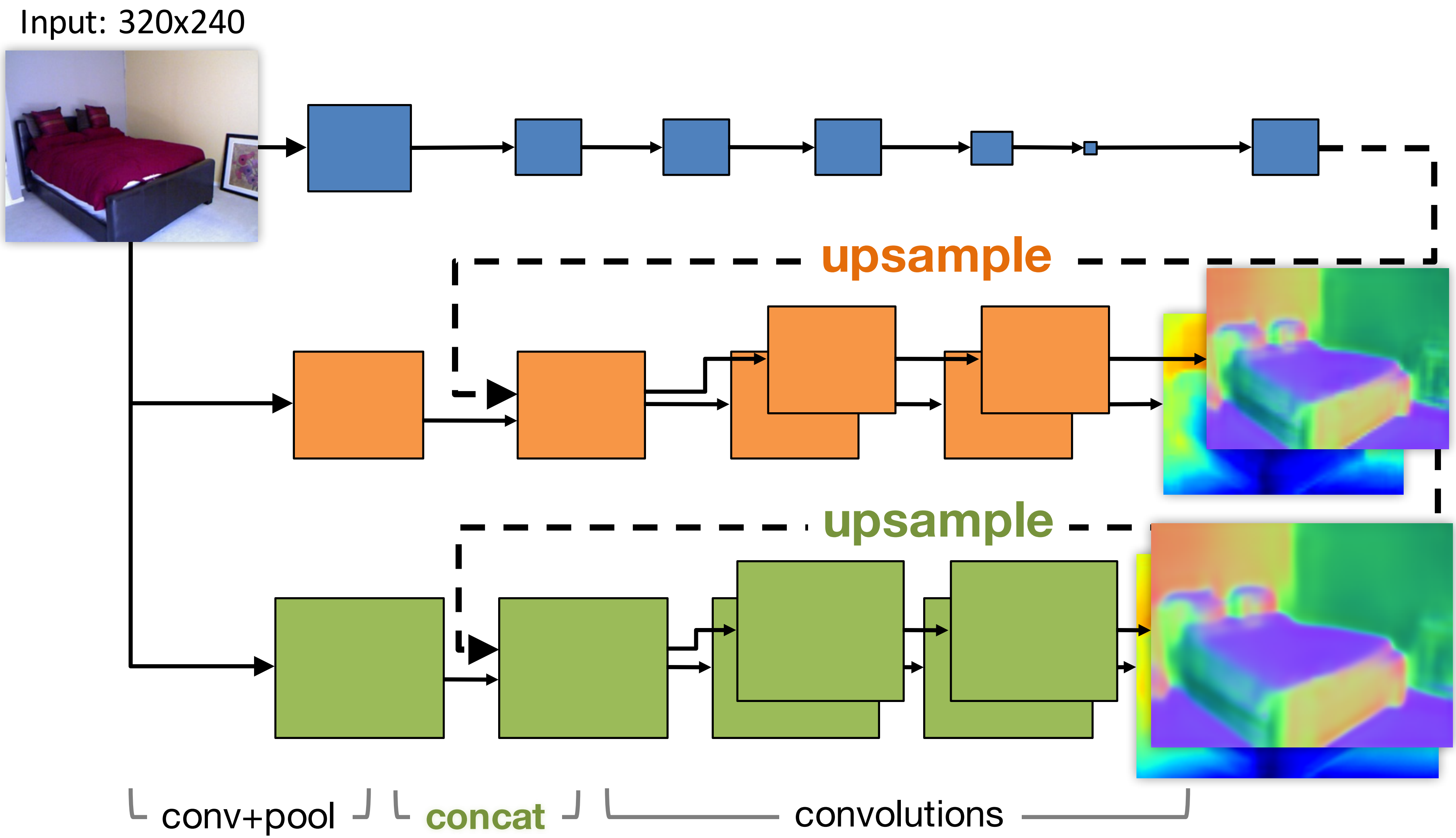
- Convnet output is per-pixel normal map

Eigen et al. architecture



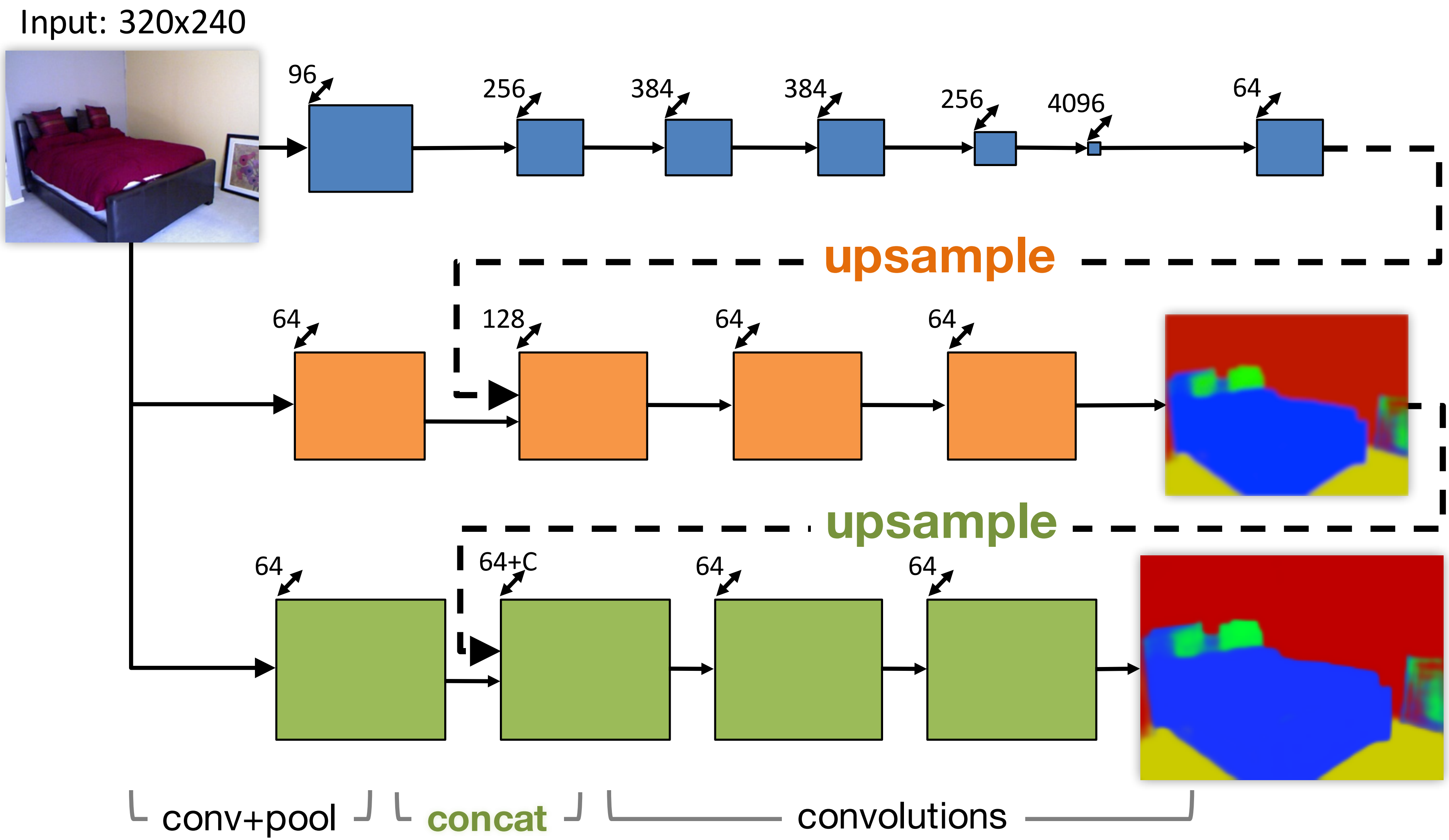
[Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture, Eigen et al., arXiv 1411.4734, 2014]

Architecture



[Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture, Eigen et al., arXiv 1411.4734, 2014]

Multi-Scale Convnets



[Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture, Eigen et al., arXiv 1411.4734, 2014]

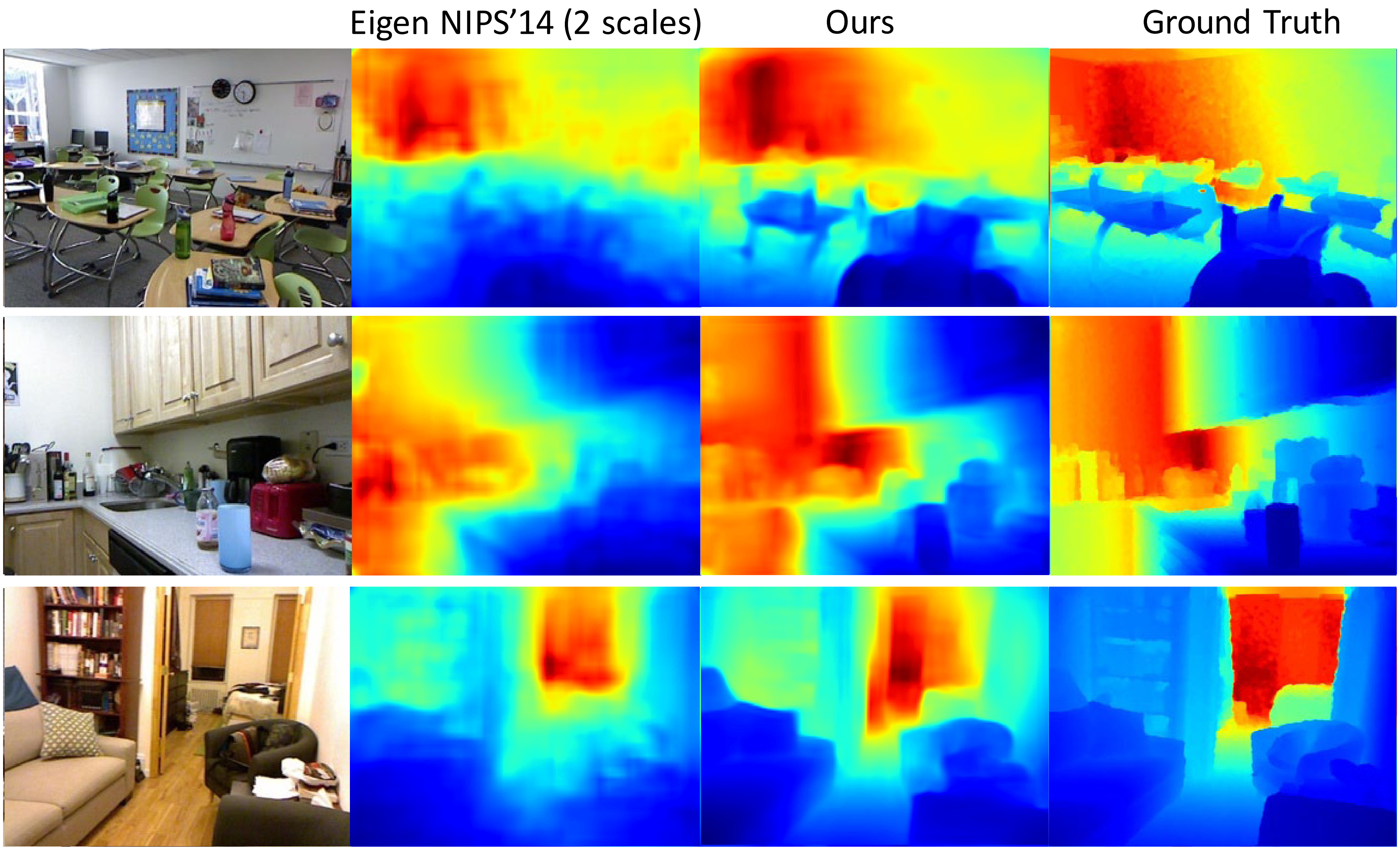
Use Appropriate Loss Functions

Depth: $d = D - D^*$ D = log predicted depth, D* = log true depth

$$L_{depth}(D, D^*) = \frac{1}{n}\sum_i d_i^2 - \frac{1}{2n^2}\left(\sum_i d_i\right)^2 + \frac{1}{n}\sum_i[(\nabla_x d_i)^2 + (\nabla_y d_i)^2]$$

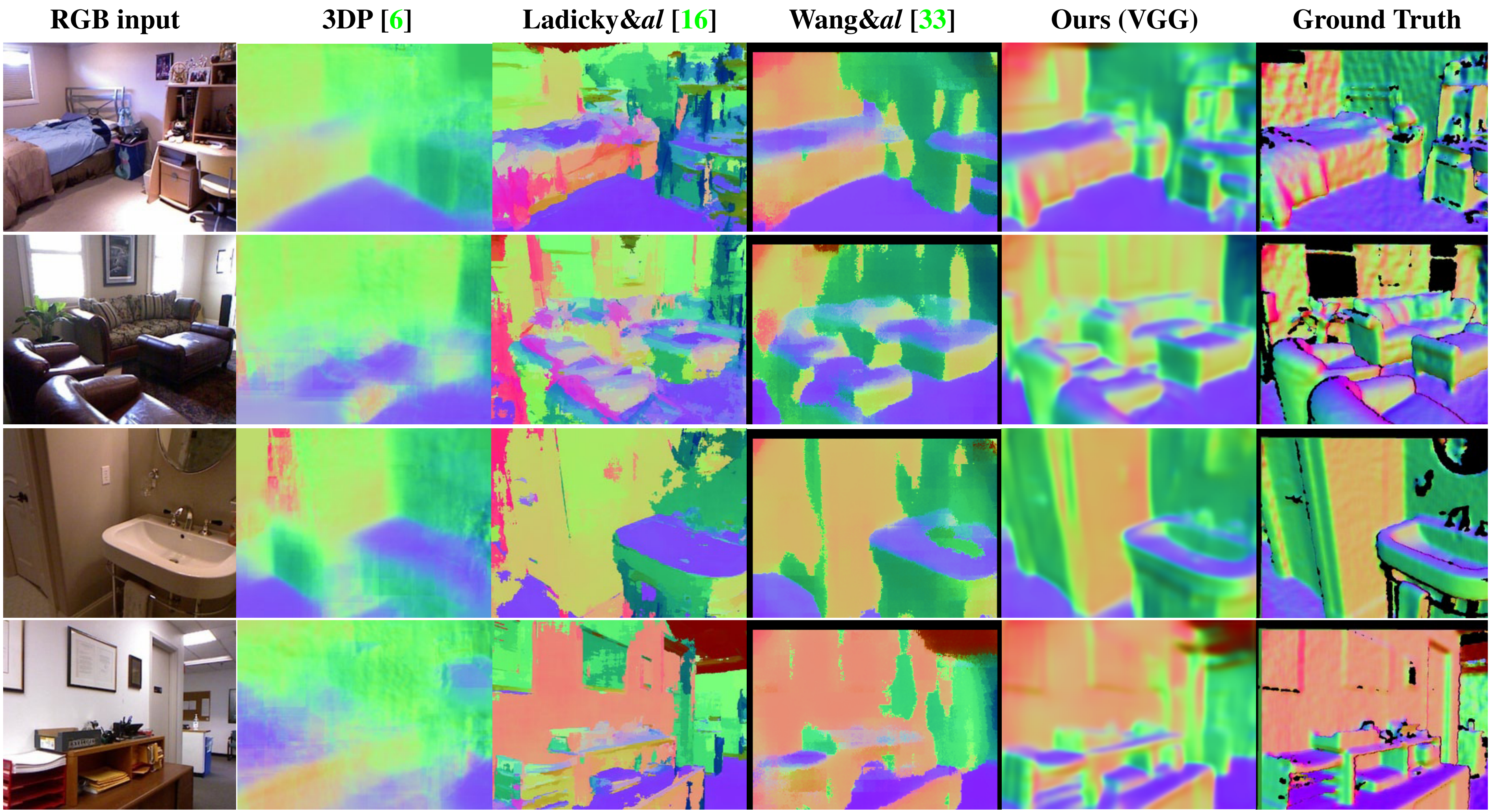
[Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture, Eigen et al., arXiv 1411.4734, 2014]

Depths Comparison



[Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture, Eigen et al., arXiv 1411.4734, 2014]

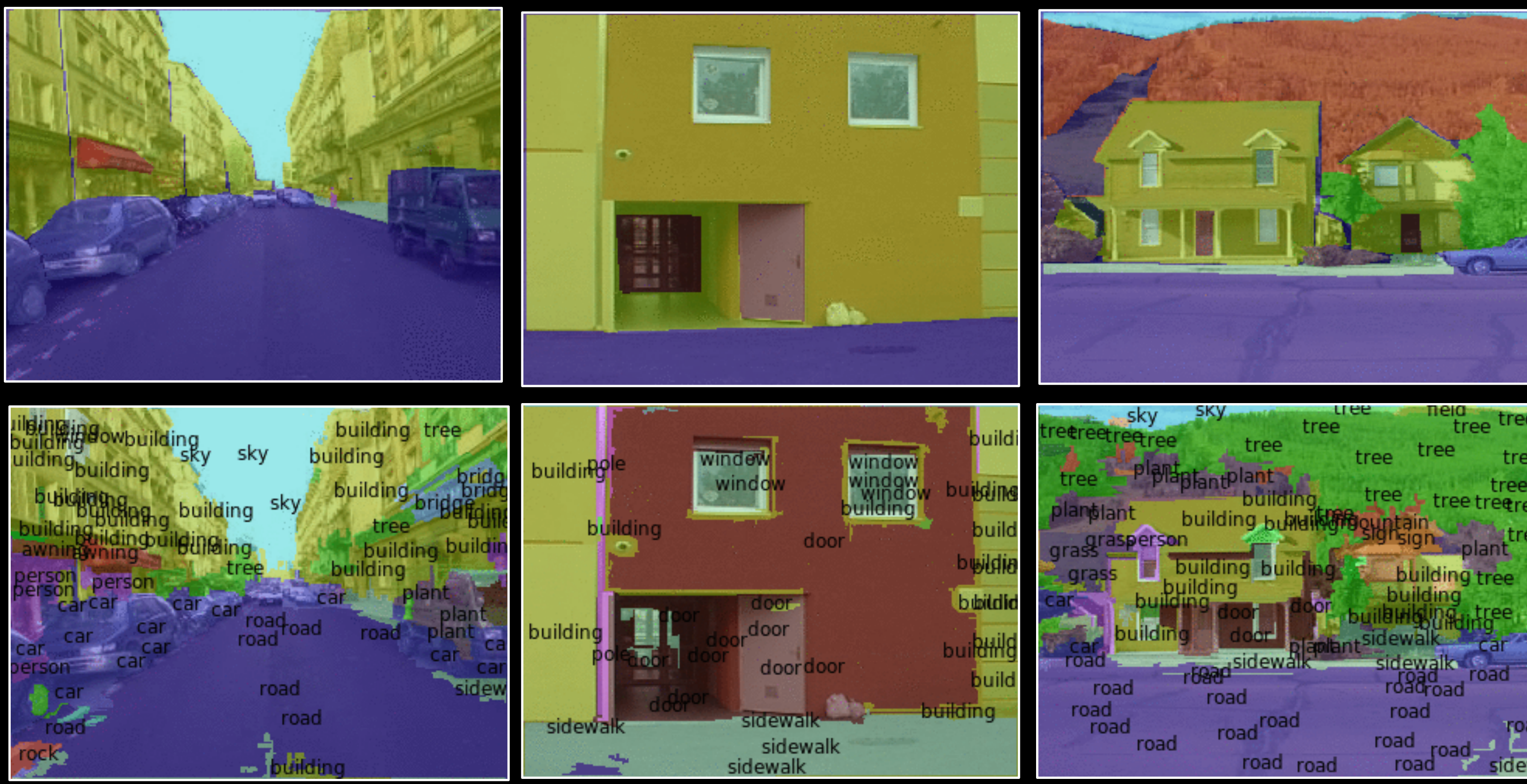
Surface Normals



[Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture, Eigen et al., arXiv 1411.4734, 2014]

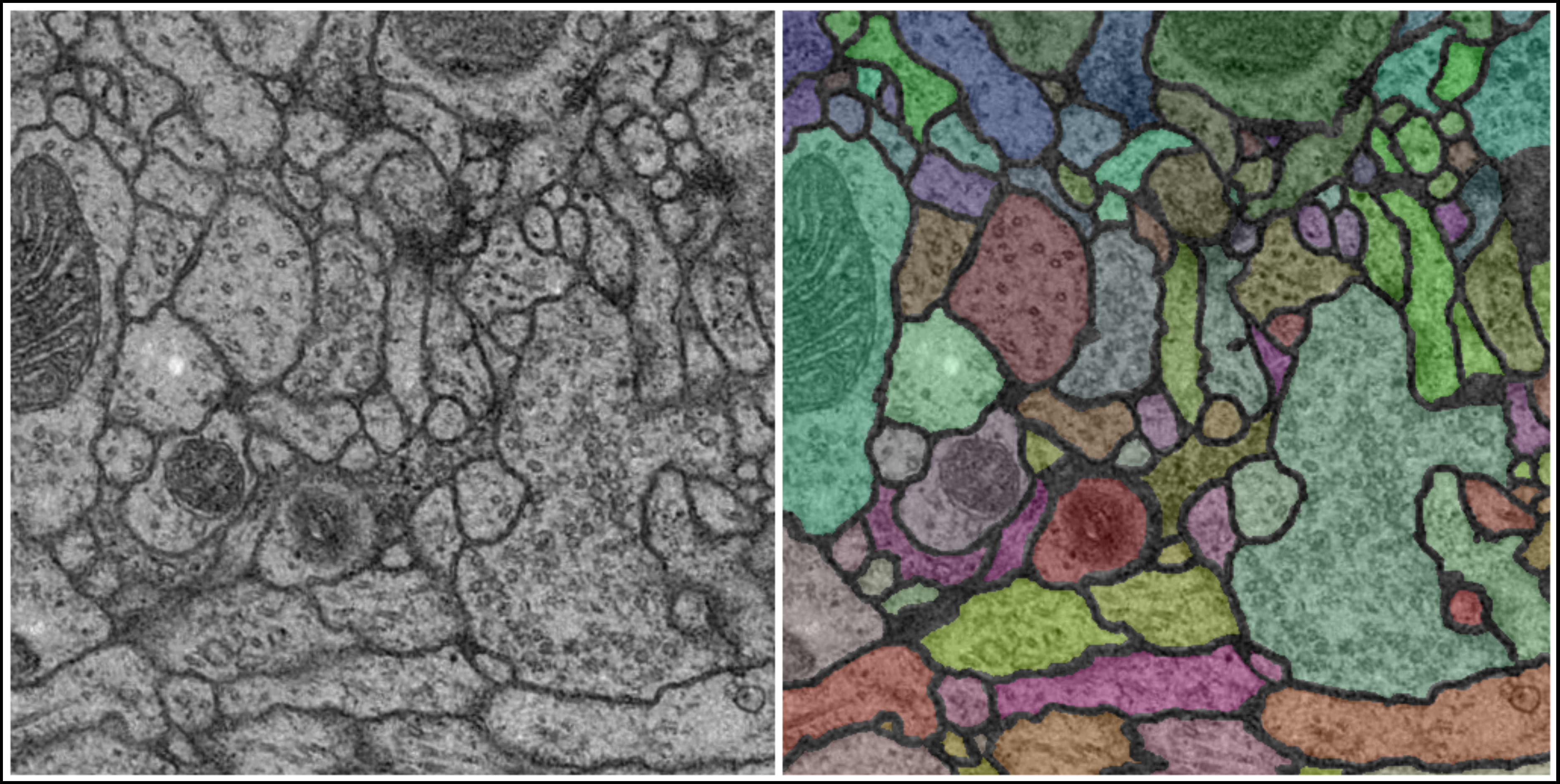
Scene Parsing

- Farabet et al. "Learning hierarchical features for scene labeling" PAMI 2013



Segmentation

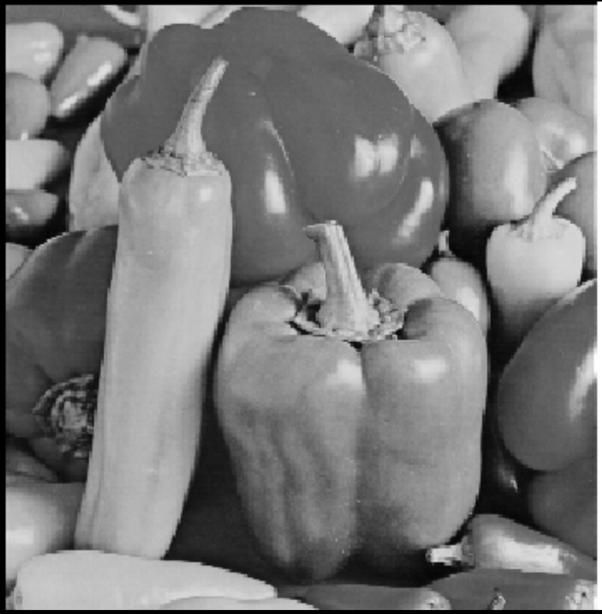
- Ciresan et al. "DNN segment neuronal membranes..." NIPS 2012
- Turaga et al. "Maximin learning of image segmentation" NIPS 2009



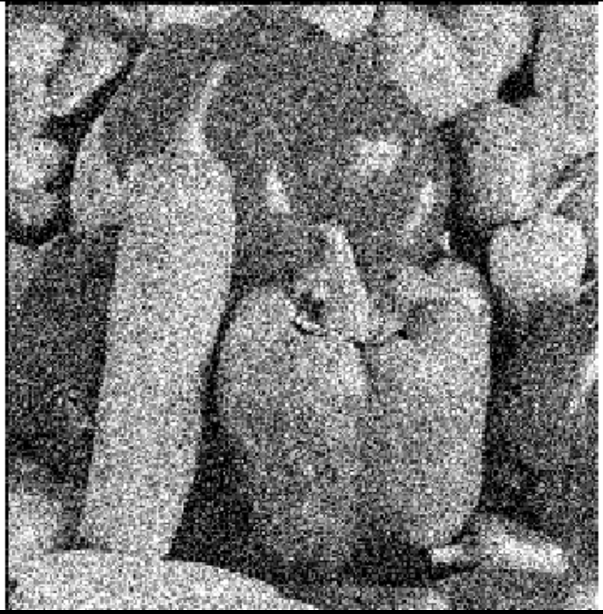
Denoising with ConvNets

- Burger et al. "Can plain NNs compete with BM3D?" CVPR 2012

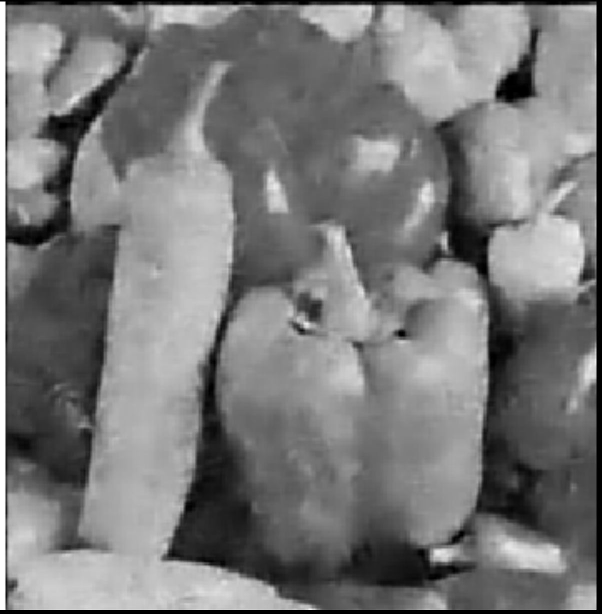
Original



Noised

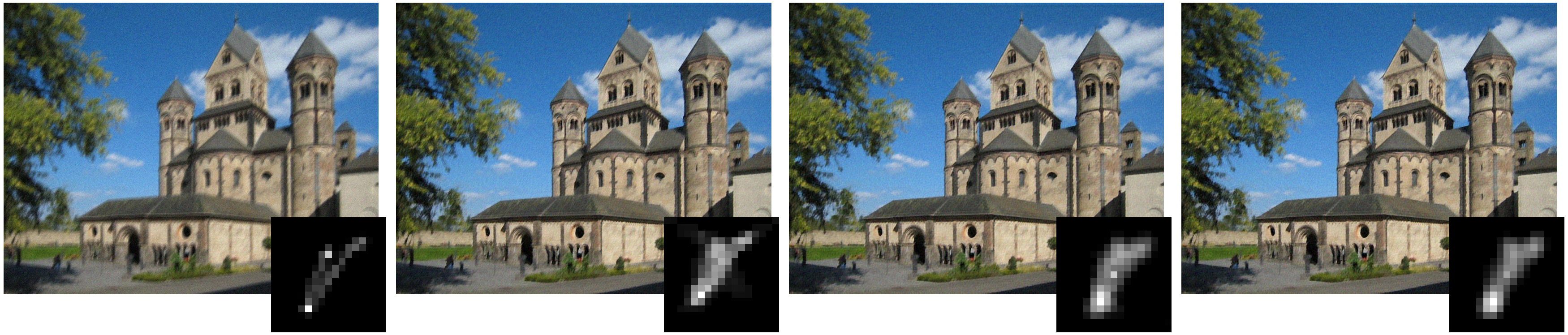


Denoised



Deblurring with Convnets

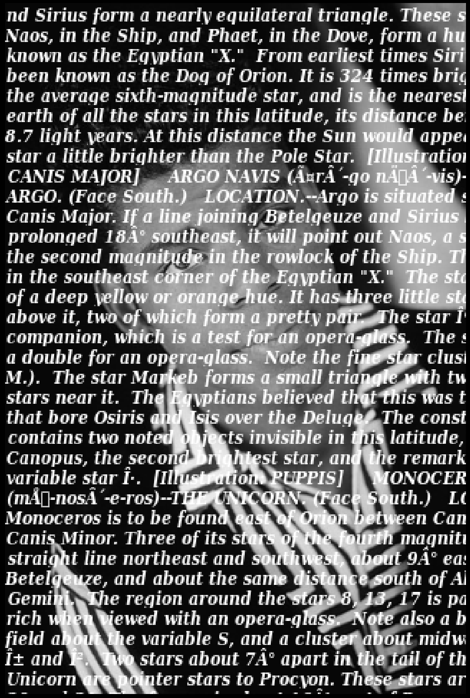
- Blind deconvolution
  - Learning to Deblur, Schuler et al., arXiv 1406.7444, 2014



Blurry image with ground truth kernel | Result of [Zho+13] PSNR 23.17 | Deblurring result w. noise *agnostic* training PSNR 23.29 | Deblurring result w. noise *specific* training **PSNR 23.41**

Inpainting with Convnets

- Image Denoising and Inpainting with Deep Neural Networks, Xie et al. NIPS 2012.
- Mask-specific inpainting with deep neural networks, Köhler et al., Pattern Recognition 2014



Original

Schmid CVPR’10

Köhler et a

‘14

Removing Local Corruption

Restoring An Image Taken Through a Window Covered with Dirt or Rain

Rain Sequence

Each frame processed independently

David Eigen, Dilip Krishnan and Rob Fergus

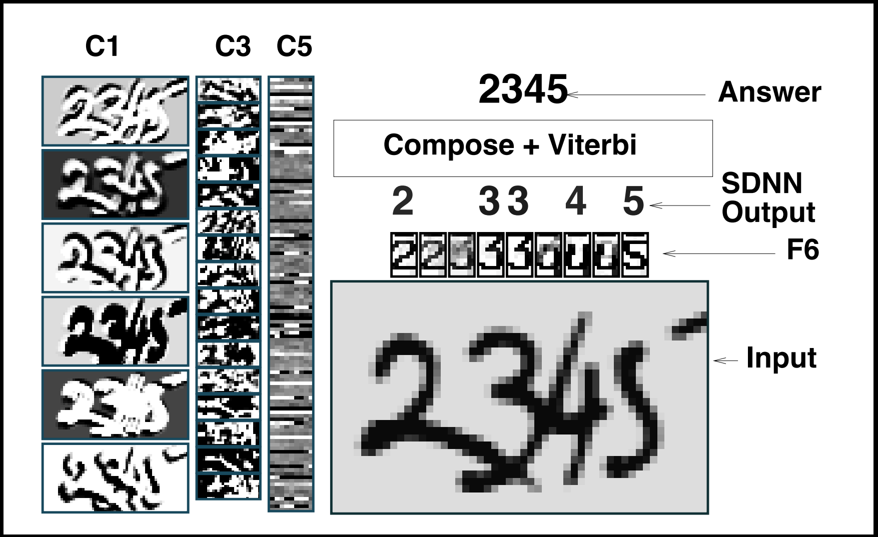
ICCV 2013

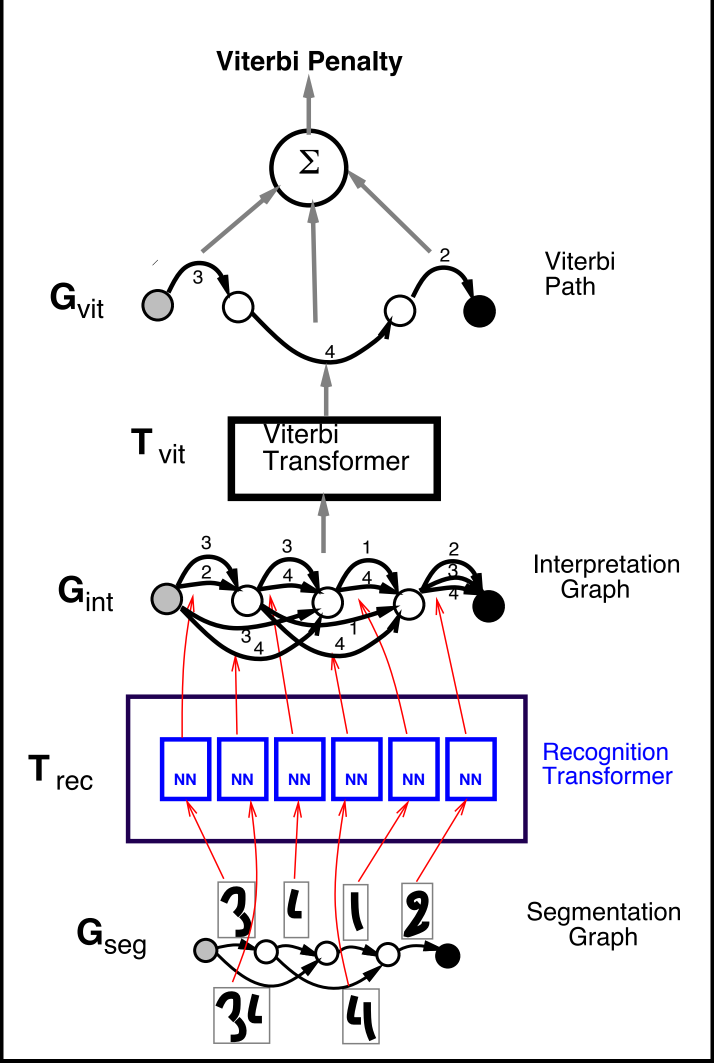
Removing Local Corruption

- Restoring An Image Taken Through a Window Covered with Dirt or Rain, Eigen et al., ICCV 2013.

Convnet + Structured Learning

- Gradient-based learning applied to document recognition, Yann LeCun, Leon Bottou, Yoshua Bengio and Patrick Haffner, Proc. IEEE, Nov 1998.





Convnet + Structured Learning

- Learning Deep Structured Models, Liang-Chieh Chen, Alexander G. Schwing, Alan L. Yuille, Raquel Urtasun, arXiv 1407.2538, 2014
- Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation, J. Tompson, A. Jain, Y. LeCun, C. Bregler, NIPS 2014
- Lots more recently……

BODY TRACKING

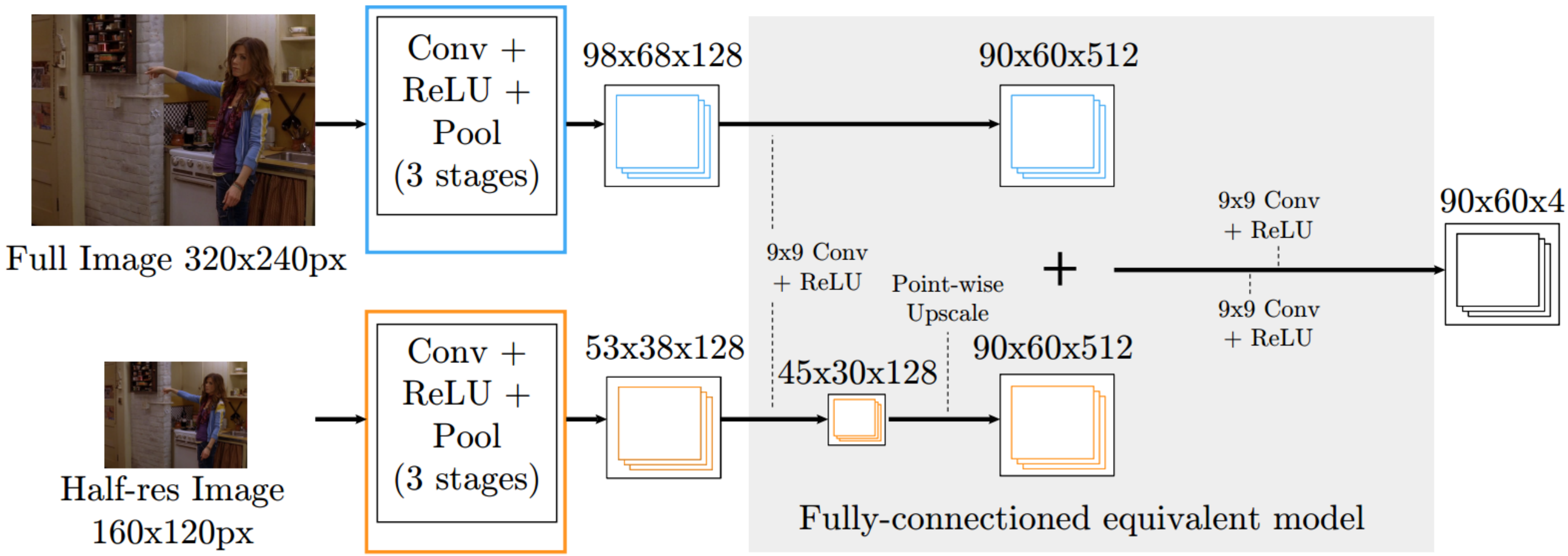
- Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation

J. Tompson, A. Jain, Y. LeCun, C. Bregler, NIPS 2014

NYU

BODY TRACKING: PART DETECTOR

Simplified multi-resolution efficient model:

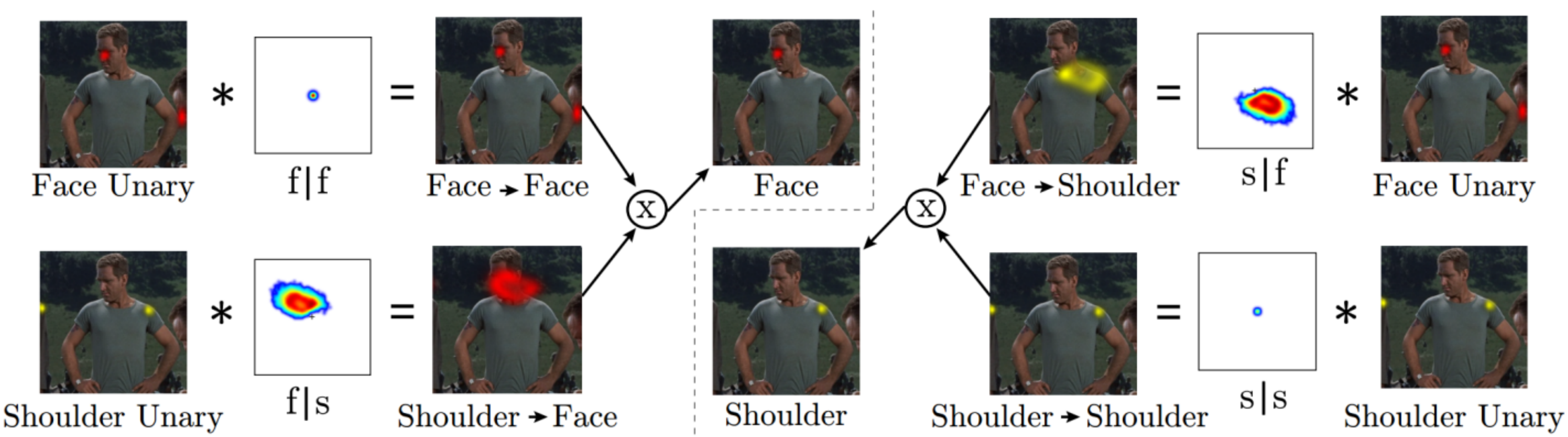


BODY TRACKING: SPATIAL MODEL

Start with MRF formulation

- “Convolutional priors”
- Sum-product belief propagation

$$\bar{p}_A = \frac{1}{Z} \prod_{v \in V} \left(p_{A|v} * p_v + b_{v \to A} \right)$$



BODY TRACKING: SPATIAL MODEL

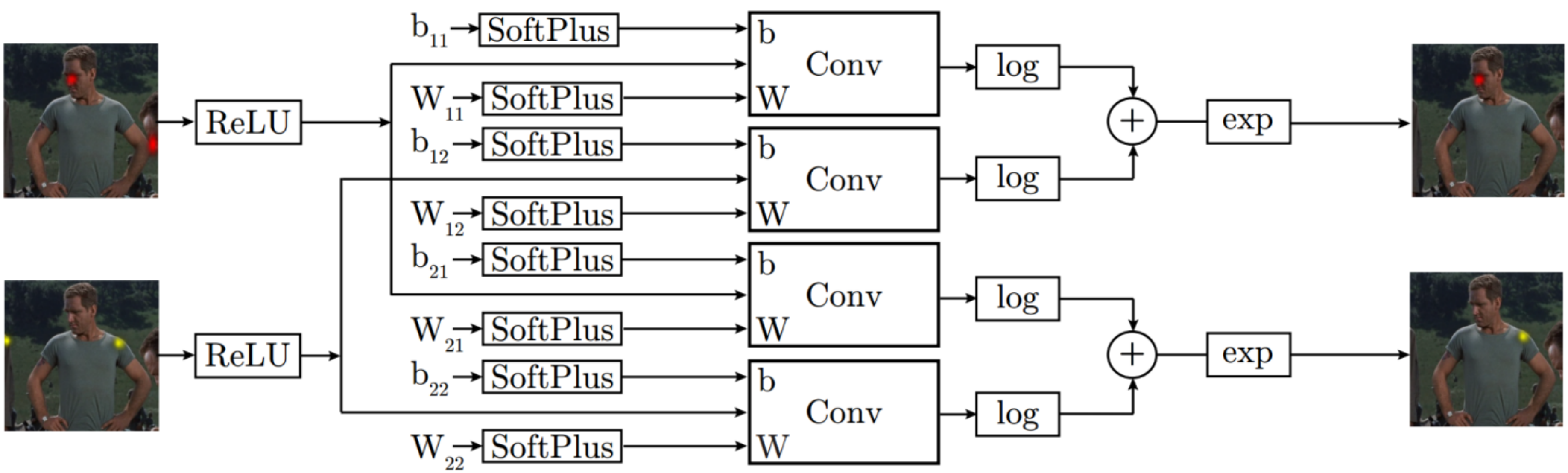
Implement it as a network (no longer MRF)!

$$\bar{p}_A = \frac{1}{Z} \prod_{v \in V} \left(p_{A|v} * p_v + b_{v \to A} \right)$$

$$\bar{e}_A = \exp\left(\sum_{v \in V} \left[\log\left(\mathrm{SoftPlus}\left(e_{A|v} \right) * \mathrm{ReLU}\left(e_v \right) + \mathrm{SoftPlus}\left(b_{v \to A} \right) \right) \right] \right)$$

$$\text{where: } \mathrm{SoftPlus}(x) = {}^1\!/\!_\beta \log\left(1 + \exp(\beta x)\right), \; {}^1\!/\!_2 \leq \beta \leq 2$$

$$\mathrm{ReLU}(x) = \max(x, \epsilon), \; 0 < \epsilon \leq 0.01$$



References

- [Slide 5]
- P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, Object Detection with Discriminatively Trained Part Based Models,IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 32, No. 9, September 2010
- Zheng Song*, Qiang Chen*, Zhongyang Huang, Yang Hua, and Shuicheng Yan. Contextualizing Object Detection and Classification. In CVPR'11. (* indicates equal contribution) [No. 1 performance in VOC'10 classification task]
- [Slide 6]
- Finding the Weakest Link in Person Detectors, D. Parikh, and C. L. Zitnick, CVPR, 2011.
- [Slide 7]
- Gehler and Nowozin, On Feature Combination for Multiclass Object Classification, ICCV'09
- [Slide 8]
- http://www.amazon.com/Vision-David-Marr/dp/0716712849
- [Slide 10]
- Yoshua Bengio and Yann LeCun: Scaling learning algorithms towards AI, in Bottou, L. and Chapelle, O. and DeCoste, D. and Weston, J. (Eds), Large-Scale Kernel Machines, MIT Press, 2007

References

- [Slide 11]
- S. Lazebnik, C. Schmid, and J. Ponce, Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories, CVPR 2006
- [Slide 12]
- Christoph H. Lampert, Hannes Nickisch, Stefan Harmeling: "Learning To Detect Unseen Object Classes by Between-Class Attribute Transfer", IEEE Computer Vision and Pattern Recognition (CVPR), Miami, FL, 2009
- [Slide 14] Riesenhuber, M. & Poggio, T. (1999). Hierarchical Models of Object Recognition in Cortex. Nature Neuroscience 2: 1019-1025.
- http://www.scholarpedia.org/article/Neocognitron
- K. Fukushima: "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position", Biological Cybernetics, 36[4], pp. 193-202 (April 1980).
- Y. LeCun, L. Bottou, Y. Bengio and P. Haffner: Gradient-Based Learning Applied to Document Recognition, Proceedings of the IEEE, 86(11):2278-2324, November 1998

References

- [Slide 30]
- Y-Lan Boureau, Jean Ponce, and Yann LeCun, A theoretical analysis of feature pooling in vision algorithms, Proc. International Conference on Machine learning (ICML'10), 2010
- [Slide 31]
- Q.V. Le, J. Ngiam, Z. Chen, D. Chia, P. Koh, A.Y. Ng , Tiled Convolutional Neural Networks. NIPS, 2010
- http://ai.stanford.edu/~quocle/TCNNweb
- Matthew D. Zeiler, Graham W. Taylor, and Rob Fergus, Adaptive Deconvolutional Networks for Mid and High Level Feature Learning, International Conference on Computer Vision(November 6-13, 2011)
- [Slide 32]
- Yuanhao Chen, Long Zhu, Chenxi Lin, Alan Yuille, Hongjiang Zhang. Rapid Inference on a Novel AND/OR graph for Object Detection, Segmentation and Parsing. NIPS 2007.

References

- [Slide 35]
- P. Smolensky, Parallel Distributed Processing: Volume 1: Foundations, D. E. Rumelhart, J. L. McClelland, Eds. (MIT Press, Cambridge, 1986), pp. 194–281.
- G. E. Hinton, Neural Comput. 14, 1711 (2002).
- [Slide 36]
- M. Ranzato, Y. Boureau, Y. LeCun. "Sparse Feature Learning for Deep Belief Networks". Advances in Neural Information Processing Systems 20 (NIPS 2007).
- [Slide 39]
- Hinton, G. E. and Salakhutdinov, R. R., Reducing the dimensionality of data with neural networks. Science, Vol. 313. no. 5786, pp. 504 - 507, 28 July 2006.
- [Slide 41]
- A. Torralba, K. P. Murphy and W. T. Freeman, Contextual Models for Object Detection using Boosted Random Fields, Adv. in Neural Information Processing Systems 17 (NIPS), pp. 1401-1408, 2005.

References

- [Slide 42]
- Ruslan Salakhutdinov and Geoffrey Hinton, Deep Boltzmann Machines, 12th International Conference on Artificial Intelligence and Statistics (2009).
- [Slide 44]
- P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, Object Detection with Discriminatively Trained Part Based Models,IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 32, No. 9, September 2010
- Long Zhu, Yuanhao Chen, Alan Yuille, William Freeman. Latent Hierarchical Structural Learning for Object Detection. CVPR 2010.
- [Slide 45]
- Matthew D. Zeiler, Graham W. Taylor, and Rob Fergus, Adaptive Deconvolutional Networks for Mid and High Level Feature Learning, International Conference on Computer Vision(November 6-13, 2011)

References

- [Slide 48]
- S.C. Zhu and D. Mumford, A Stochastic Grammar of Images, Foundations and Trends in Computer Graphics and Vision, Vol.2, No.4, pp 259-362, 2006.
- [Slide 49]
- R. Girshick, P. Felzenszwalb, D. McAllester, Object Detection with Grammar Models, NIPS 2011
- [Slide 50]
- P. Felzenszwalb, D. Huttenlocher, Pictorial Structures for Object Recognition, International Journal of Computer Vision, Vol. 61, No. 1, January 2005
- M. Fischler and R. Elschlager. The Representation and Matching of Pictoral Structures. (1973)
- [Slide 51]
- S. Fidler, M. Boben, A. Leonardis. A coarse-to-fine Taxonomy of Constellations for Fast Multi-class Object Detection. ECCV 2010.
- S. Fidler and A. Leonardis. Towards Scalable Representations of Object Categories: Learning a Hierarchy of Parts. CVPR 2007.

References

- [Slide 52]
- Long Zhu, Chenxi Lin, Haoda Huang, Yuanhao Chen, Alan Yuille. Unsupervised Structure Learning: Hierarchical Recursive Composition, Suspicious Coincidence and Competitive Exclusion. ECCV 2008.
- [Slide 53]
- Hinton, G. E., Krizhevsky, A. and Wang, S, Transforming Auto-encoders. ICANN-11: International Conference on Artificial Neural Networks, 2011
- Matthew D. Zeiler, Graham W. Taylor, and Rob Fergus, Adaptive Deconvolutional Networks for Mid and High Level Feature Learning, International Conference on Computer Vision(November 6-13, 2011)
- [Slide 54]
- Q.V. Le, M.A. Ranzato, R. Monga, M. Devin, K. Chen, G.S. Corrado, J. Dean, A.Y. Ng., Building high-level features using large scale unsupervised learning. ICML, 2012.
- [Slide 55]
- Ruslan Salakhutdinov and Geoffrey Hinton, Deep Boltzmann Machines, 12th International Conference on Artificial Intelligence and Statistics (2009).

References

- [Slide 56]
- http://www.image-net.org/challenges/LSVRC/2010/
- Q.V. Le, M.A. Ranzato, R. Monga, M. Devin, K. Chen, G.S. Corrado, J. Dean, A.Y. Ng., Building high-level features using large scale unsupervised learning. ICML, 2012.
- Q.V. Le, W.Y. Zou, S.Y. Yeung, A.Y. Ng., Learning hierarchical spatio-temporal features for action recognition with independent subspace analysis, CVPR 2011