Introduction to Convolutional Networks

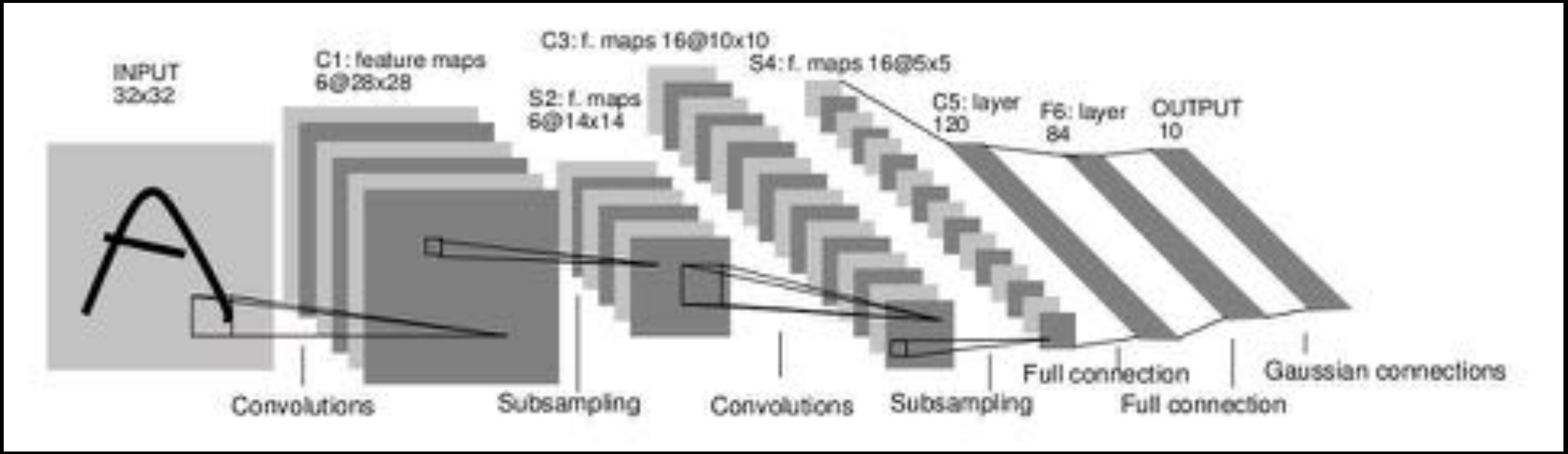
Lecture 3

Rob Fergus

New York University

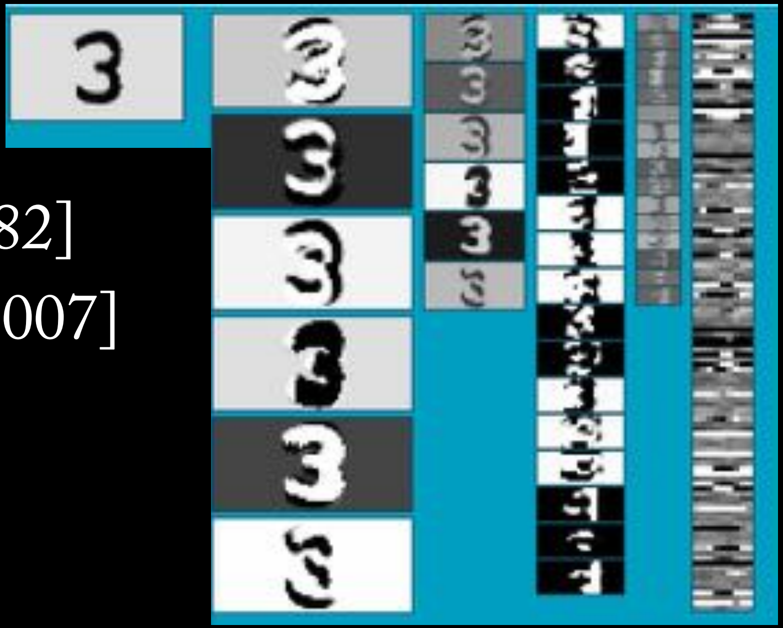
Convolutional Neural Networks

- 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



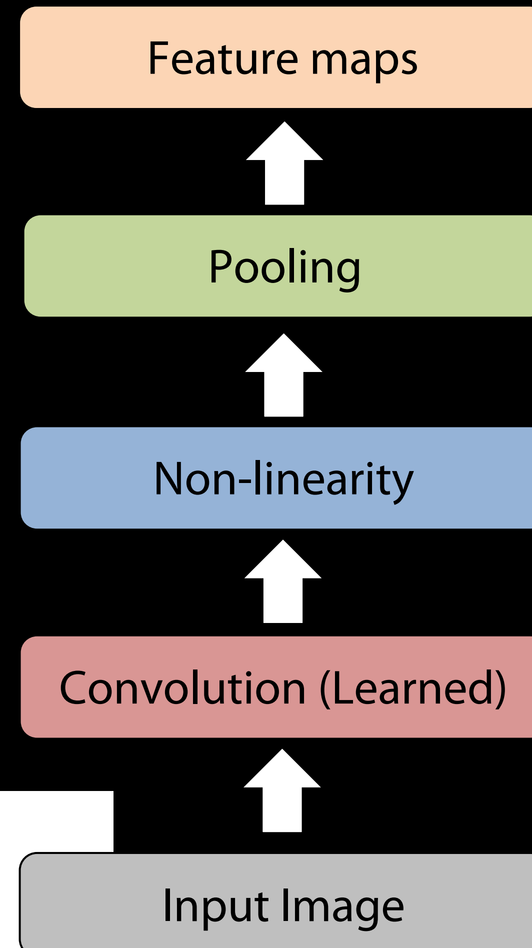
History:

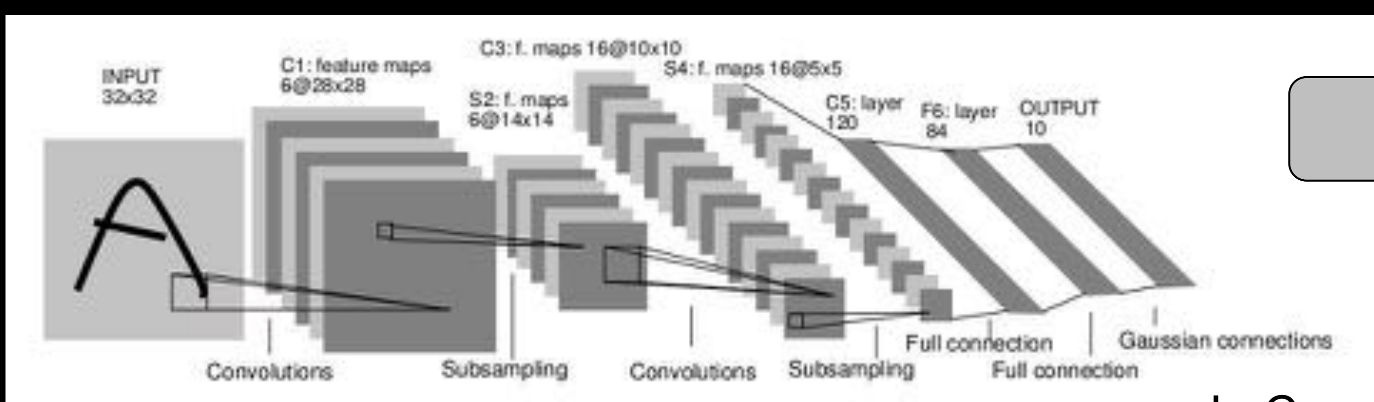
- Neocognitron [Fukushima 1971-1982]
- Convolutional Nets [LeCun 1988-2007]
- HMAX [Poggio 2002-2006]
- Many others….

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





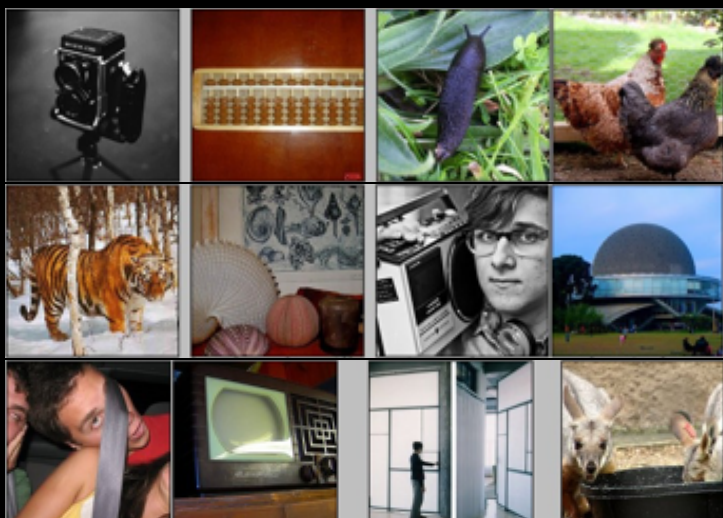
Convnet Successes

- 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)

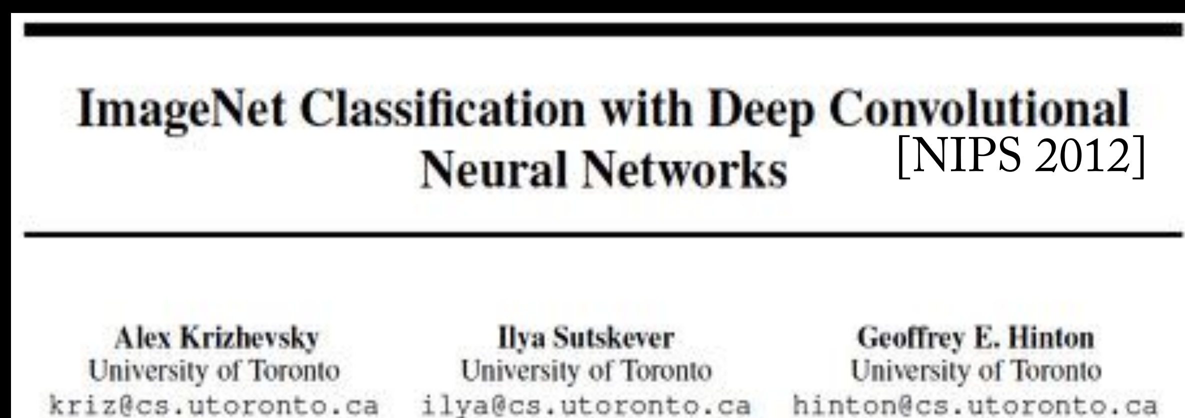
Application to ImageNet

IMAGENET



[Deng et al. CVPR 2009]

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



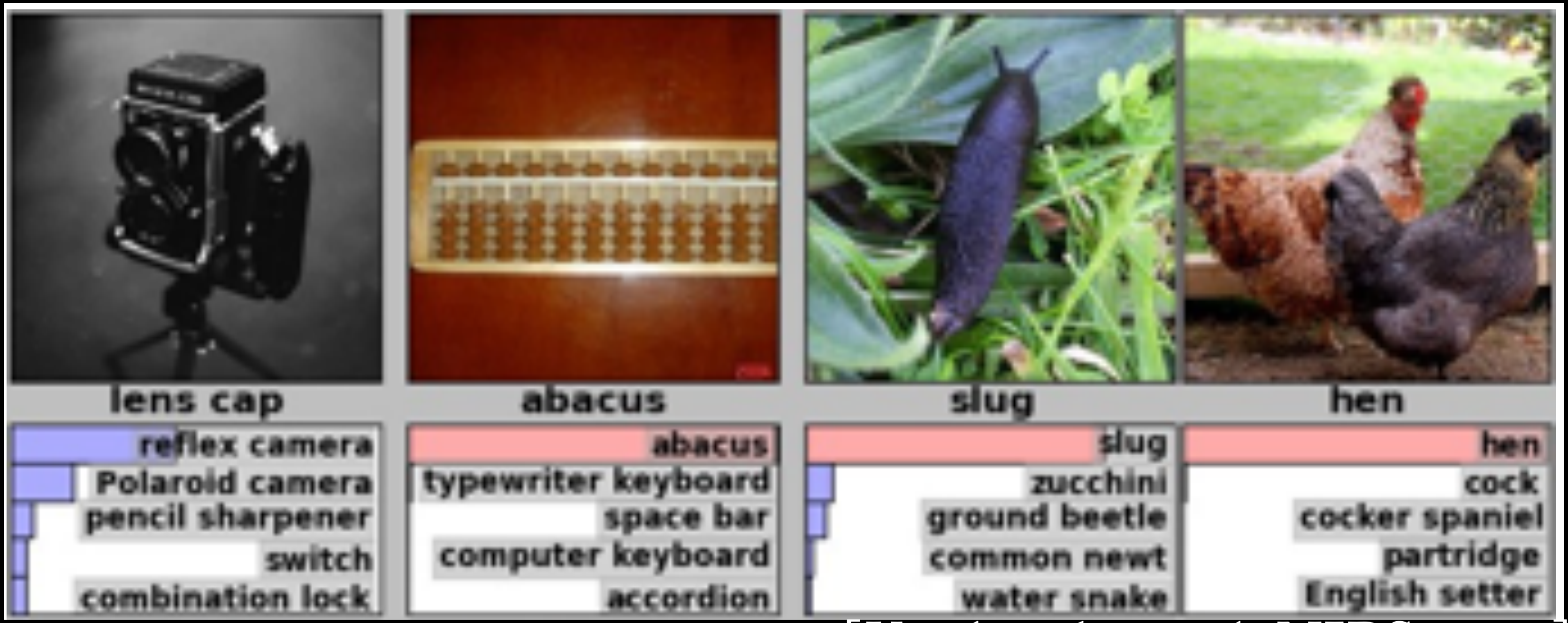
ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky	Ilya Sutskever	Geoffrey E. Hinton
University of Toronto	University of Toronto	University of Toronto
kriz@cs.utoronto.ca	ilya@cs.utoronto.ca	hinton@cs.utoronto.ca

[NIPS 2012]

Goal

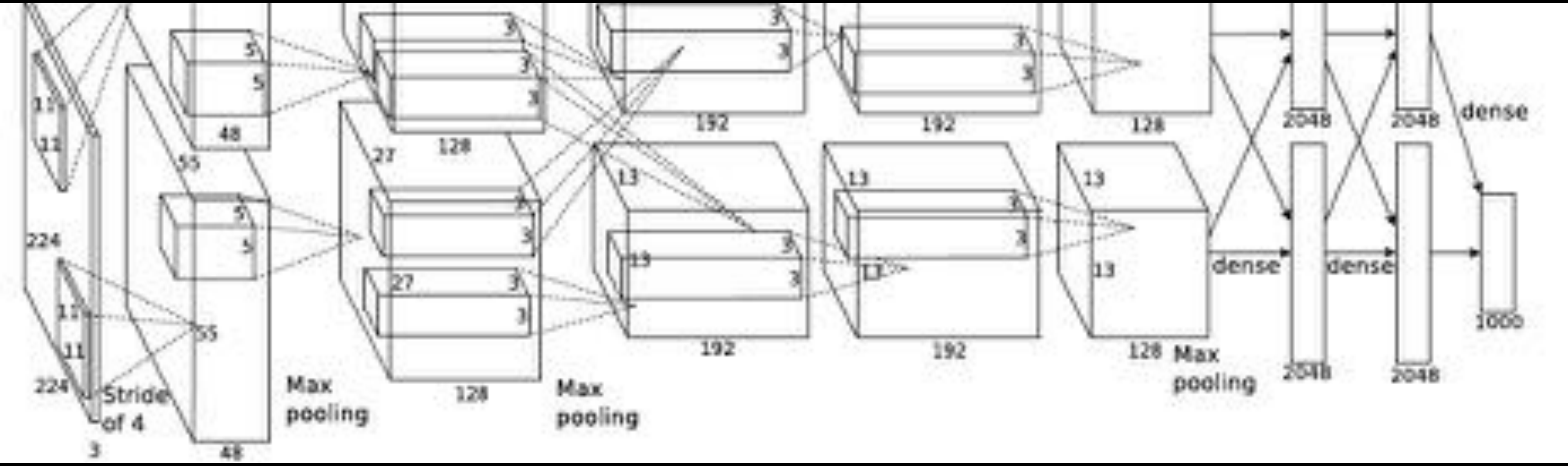
- Image Recognition
  - Pixels → Class Label



[Krizhevsky et al. NIPS 2012]

Krizhevsky et al. [NIPS2012]

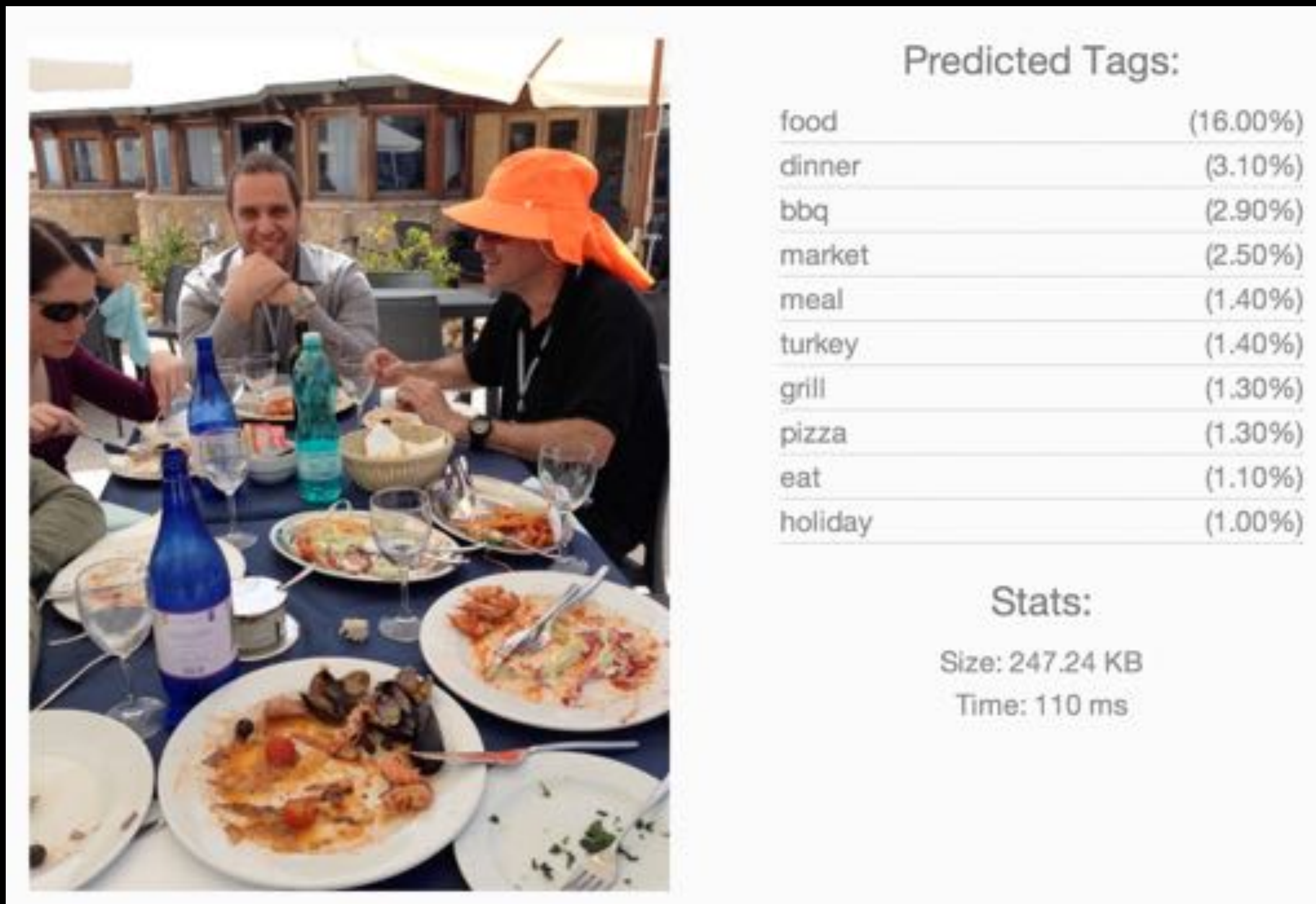
- Same model as LeCun’98 but:
  - Bigger model (8 layers)
  - More data (10^6 vs 10^3 images)
  - GPU implementation (50x speedup over CPU)
  - Better regularization (DropOut)



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

Examples

- From Clarifai.com



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

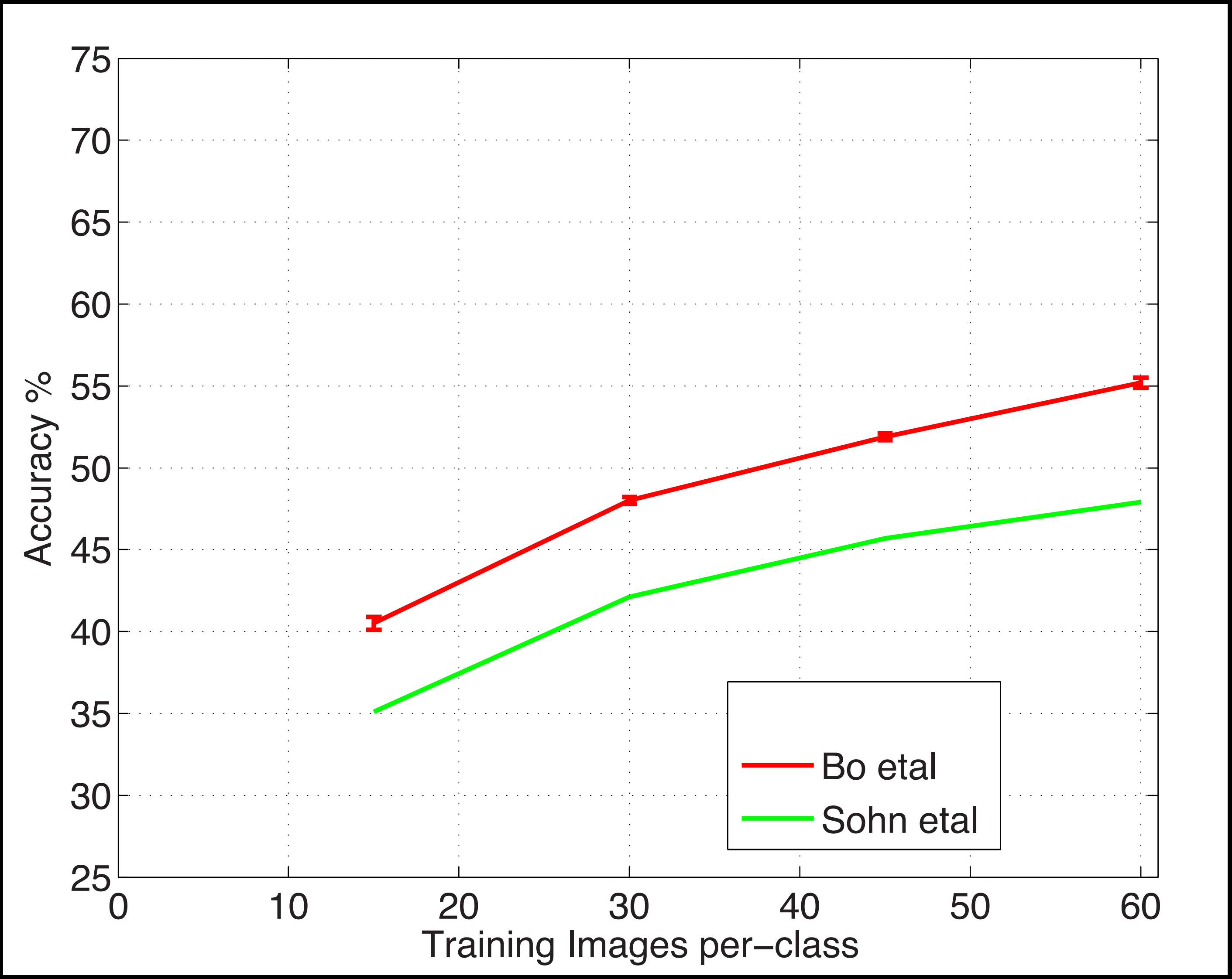
Time: 113 ms

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

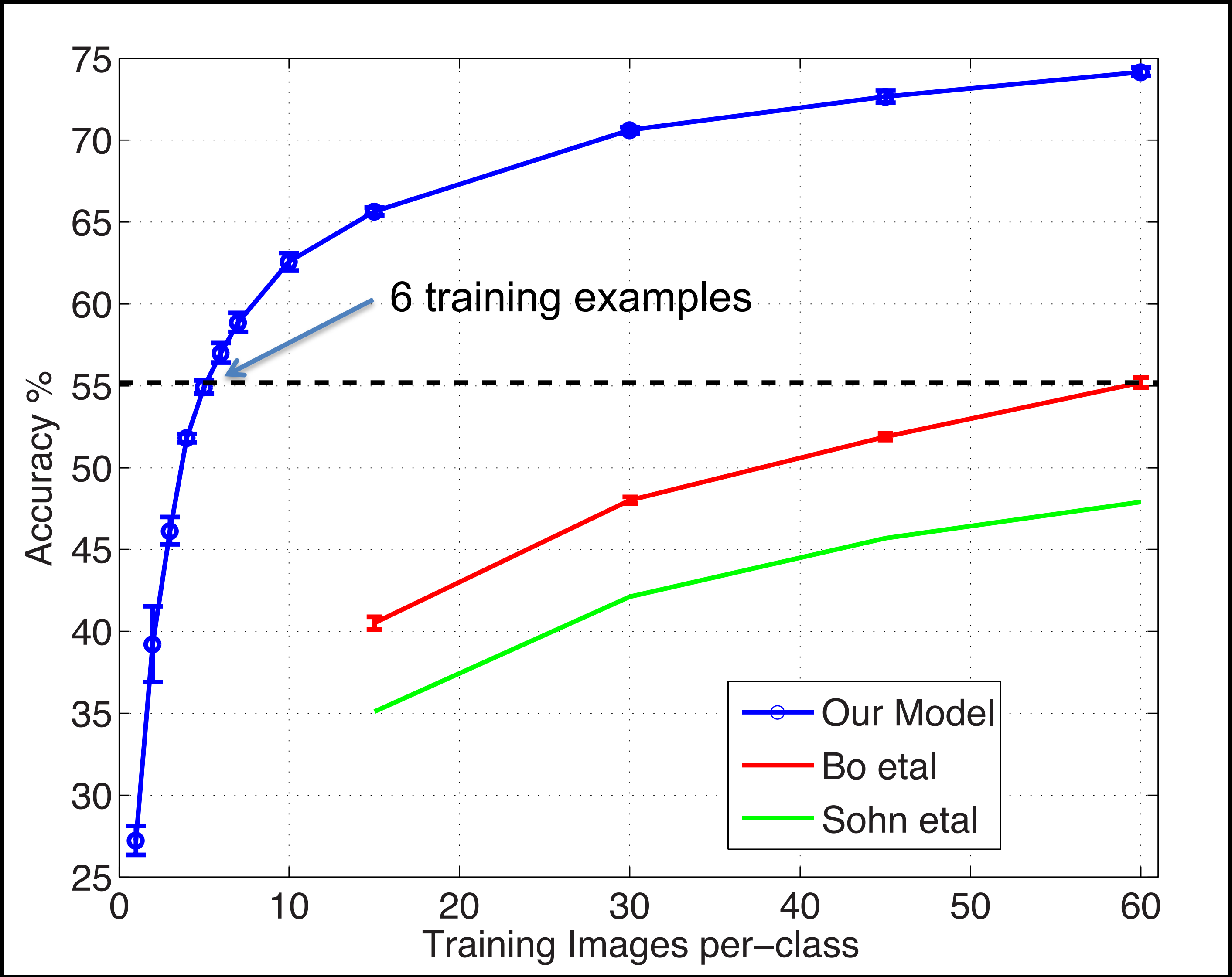
Caltech 256

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



Caltech 256

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



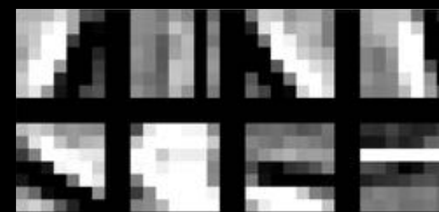
The Details

- Operations in each layer
- Architecture
- Training
- Results

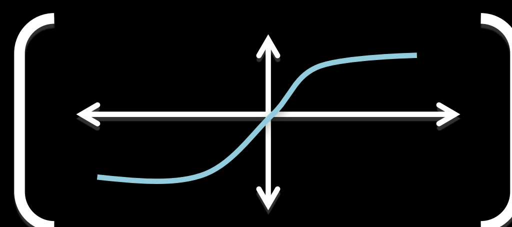
Components of Each Layer

Pixels / Features

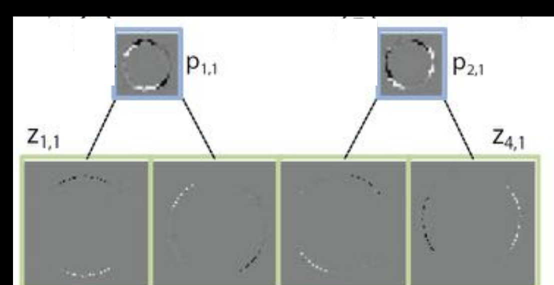
Filter with learned dictionary



Non-linearity



Spatial local max pooling

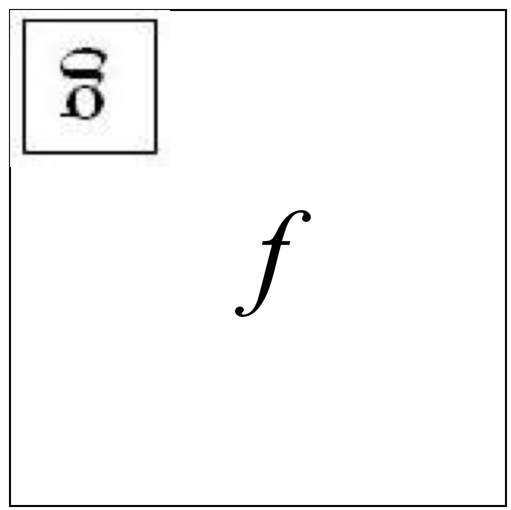


Output Features

Defining Convolution

- Let *f* be the image and *g* be the kernel. The output of convolving *f* with *g* is denoted *f* * *g*.

$$(f * g)[m,n] = \sum_{k,l} f[m-k, n-l]\, g[k,l]$$



- Convention: kernel is “flipped”
- MATLAB: conv2 (also imfilter)

Source: F. Durand

Key properties

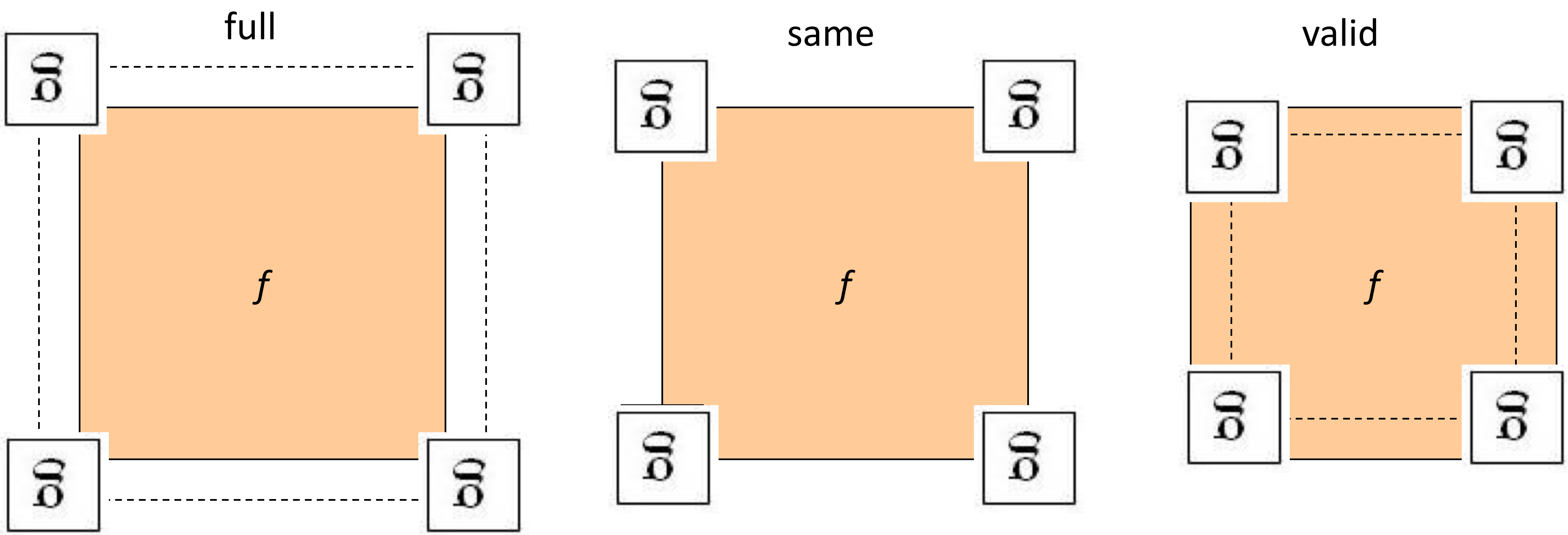
- **Linearity:** filter(f_1 + f_2) = filter(f_1) + filter(f_2)

- **Shift invariance:** same behavior regardless of pixel location: filter(shift(f)) = shift(filter(f))

- Theoretical result: any linear shift-invariant operator can be represented as a convolution

Annoying details

- What is the size of the output?
- MATLAB: conv2(f, g,*shape*)
  - *shape* = 'full' : output size is sum of sizes of f and g
  - *shape* = 'same' : output size is same as f
  - *shape* = 'valid' : output size is difference of sizes of f and g



ConvNet Architecture

- Exploits two properties of images:

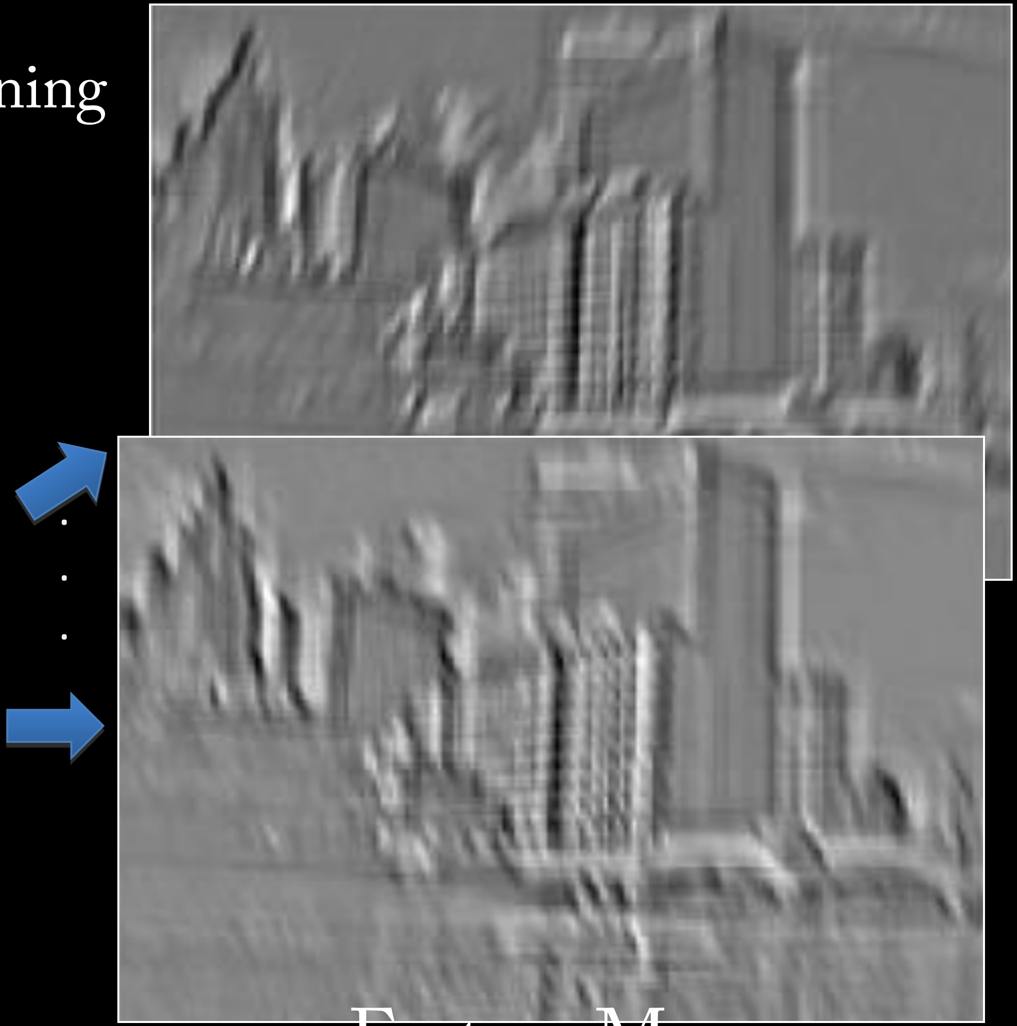
- 1. Dependencies are local
  - No need to have each unit connect to every pixel

- 2. Spatially stationary statistics
  - Translation invariant dependencies
  - Only approximately true

Filtering

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

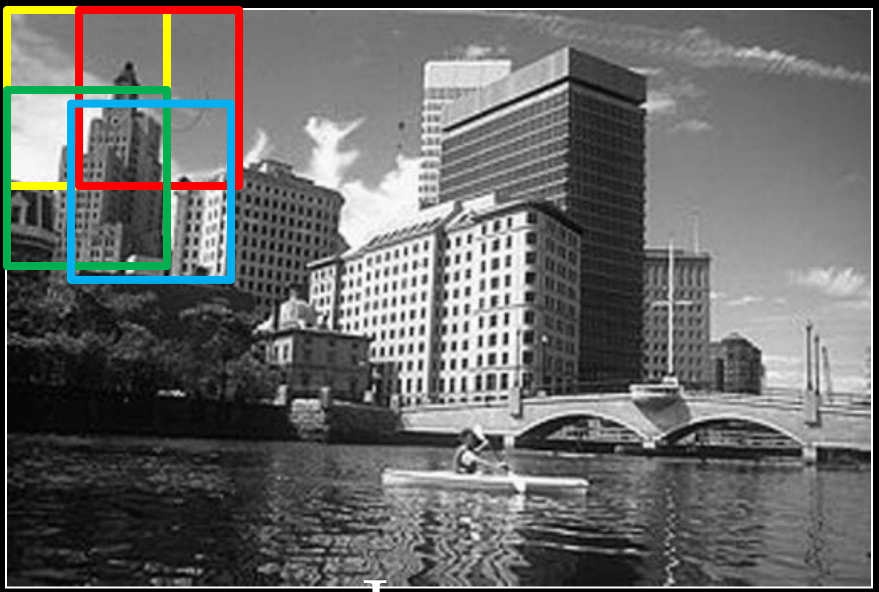
Input



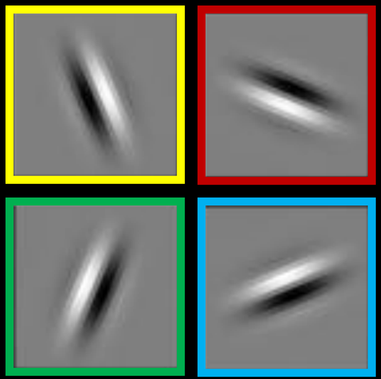
Feature Map

Filtering

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

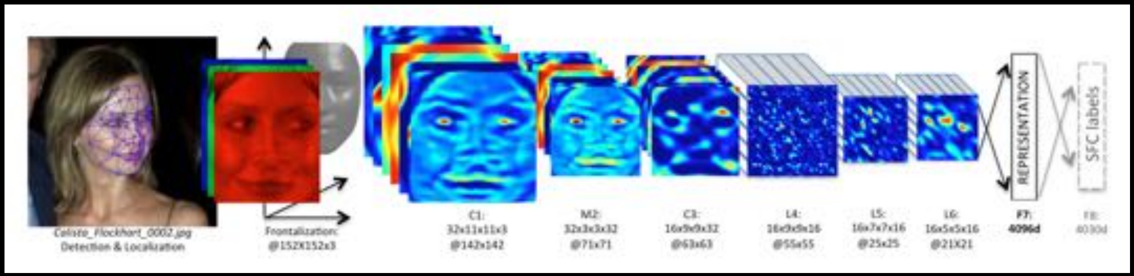


Input



Filters

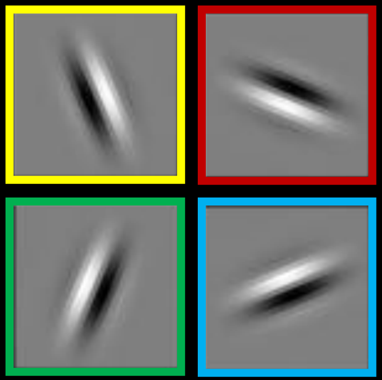
- E.g. face recognition



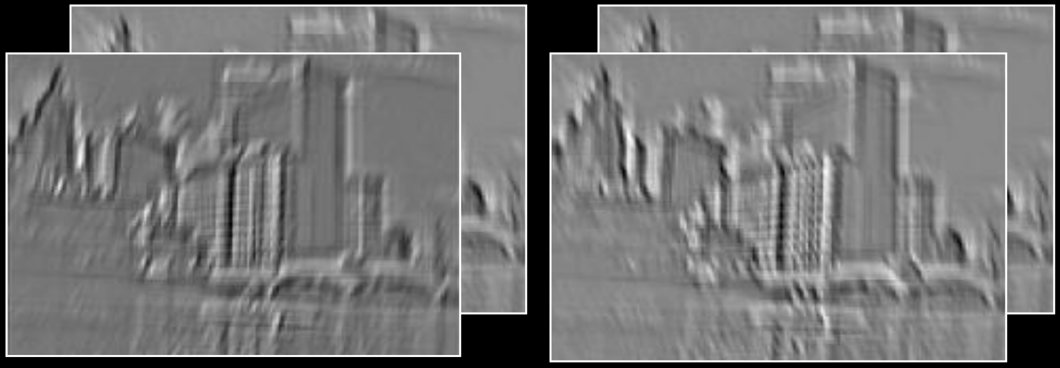
Filtering

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

Input



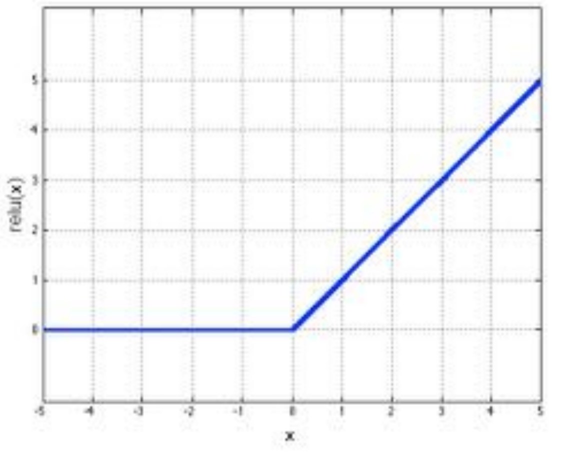
Filters



Feature maps

Non-Linearity

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

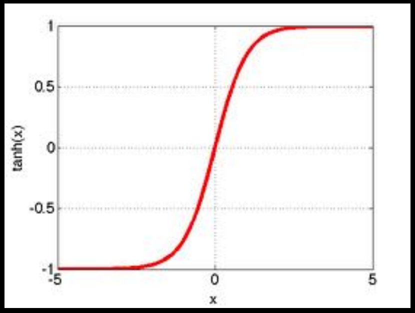


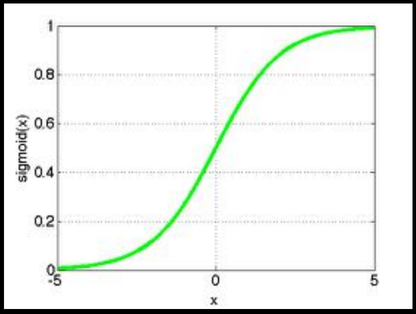
Input feature map

Output feature map

Non-Linearity

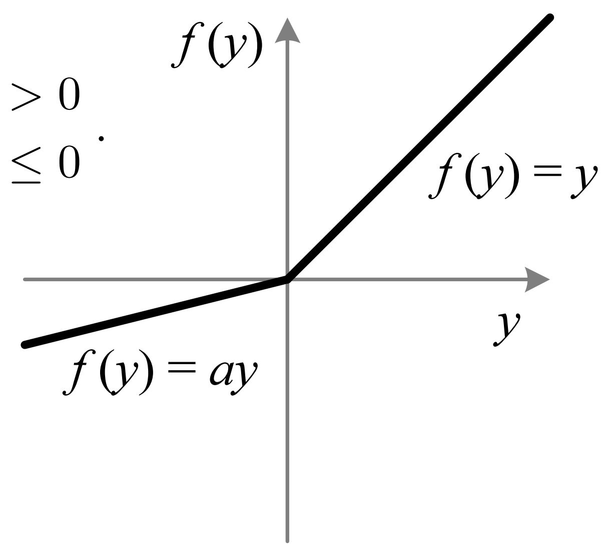
- Other choices:
  - Tanh
  - Sigmoid: 1/(1+exp(-x))
  - PReLU





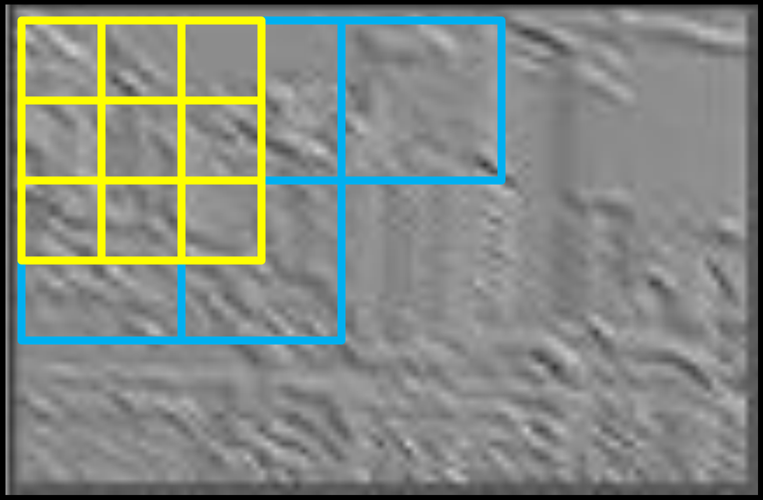
[Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification, Kaiming He et al. arXiv:1502.01852v1.pdf, Feb 2015]

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

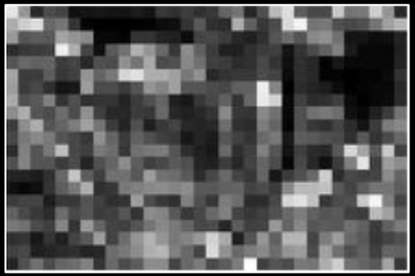


Pooling

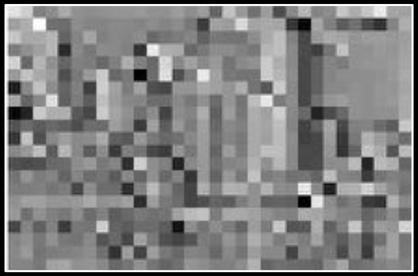
- Spatial Pooling
  - Non-overlapping / overlapping regions
  - Sum or max
  - Boureau et al. ICML'10 for theoretical analysis



Max

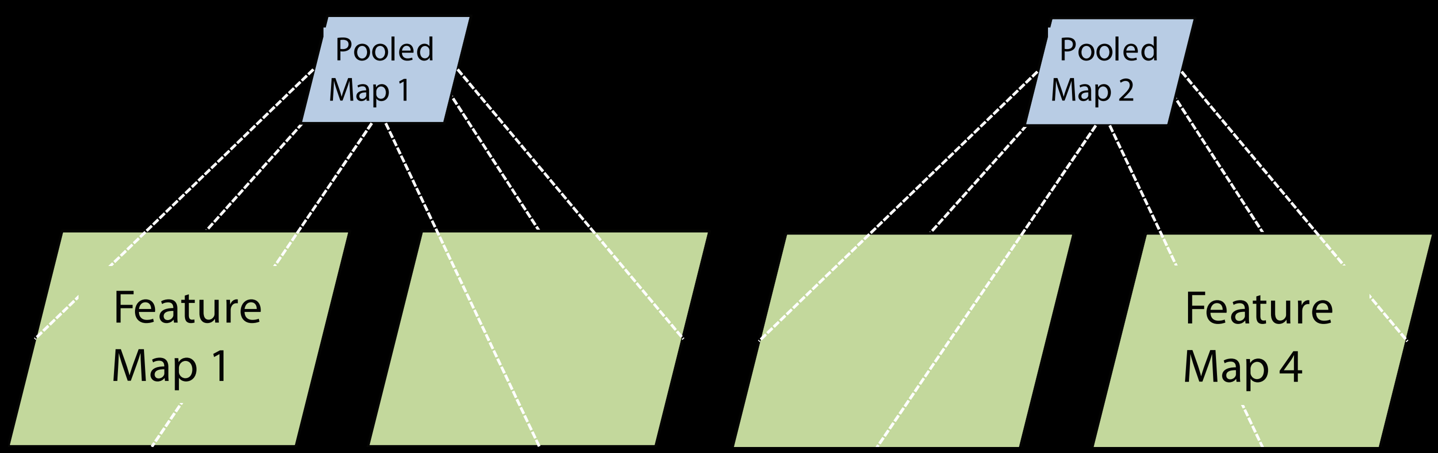


Sum



Pooling

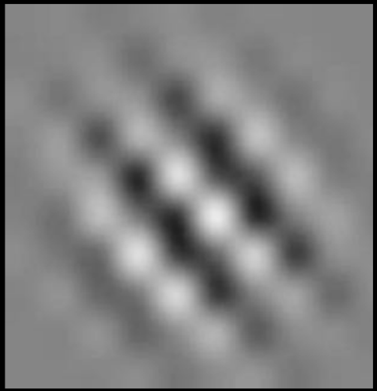
- Pooling across feature groups
  - Additional form of inter-feature competition
  - MaxOut Networks [Goodfellow et al. ICML 2013]

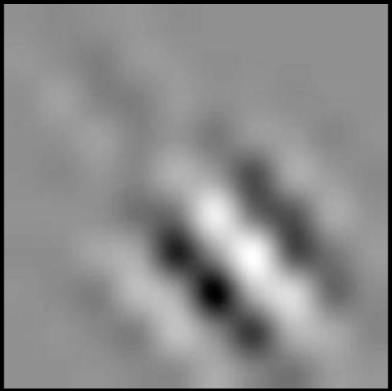


Role of Pooling

- 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

Alternative to Pooling

- Replace pooling with strided convolution
  - i.e. filters applied every r pixels (r>1)
  - [Striving for Simplicity: the all Convolutional Net, Spingenberg et al. ICL 2015]

Model		
Strided-CNN-C	ConvPool-CNN-C	All-CNN-C
Input 32×32 RGB image		
3×3 conv. 96 ReLU 3×3 conv. 96 ReLU with stride $r = 2$	3×3 conv. 96 ReLU 3×3 conv. 96 ReLU 3×3 conv. 96 ReLU	3×3 conv. 96 ReLU 3×3 conv. 96 ReLU
	3×3 max-pooling stride 2	3×3 conv. 96 ReLU with stride $r = 2$
3×3 conv. 192 ReLU 3×3 conv. 192 ReLU with stride $r = 2$	3×3 conv. 192 ReLU 3×3 conv. 192 ReLU 3×3 conv. 192 ReLU	3×3 conv. 192 ReLU 3×3 conv. 192 ReLU
	3×3 max-pooling stride 2	3×3 conv. 192 ReLU with stride $r = 2$

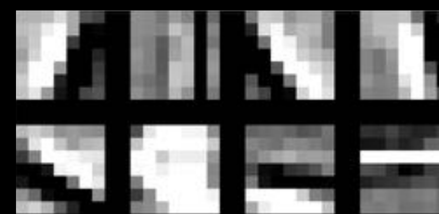
Model C	9.74%	$\approx$ 1.3 M
Strided-CNN-C	10.19%	$\approx$ 1.3 M
ConvPool-CNN-C	9.31%	$\approx$ 1.4 M
ALL-CNN-C	**9.08%**	$\approx$ 1.4 M

CIFAR-10 classification error

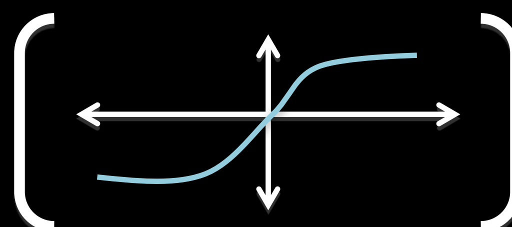
Components of Each Layer

Pixels /
Features

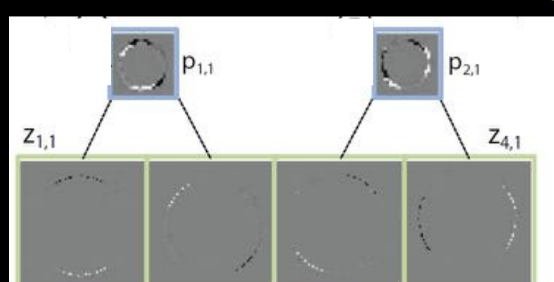
Filter with
learned dictionary



Non-linearity



Spatial local
max pooling



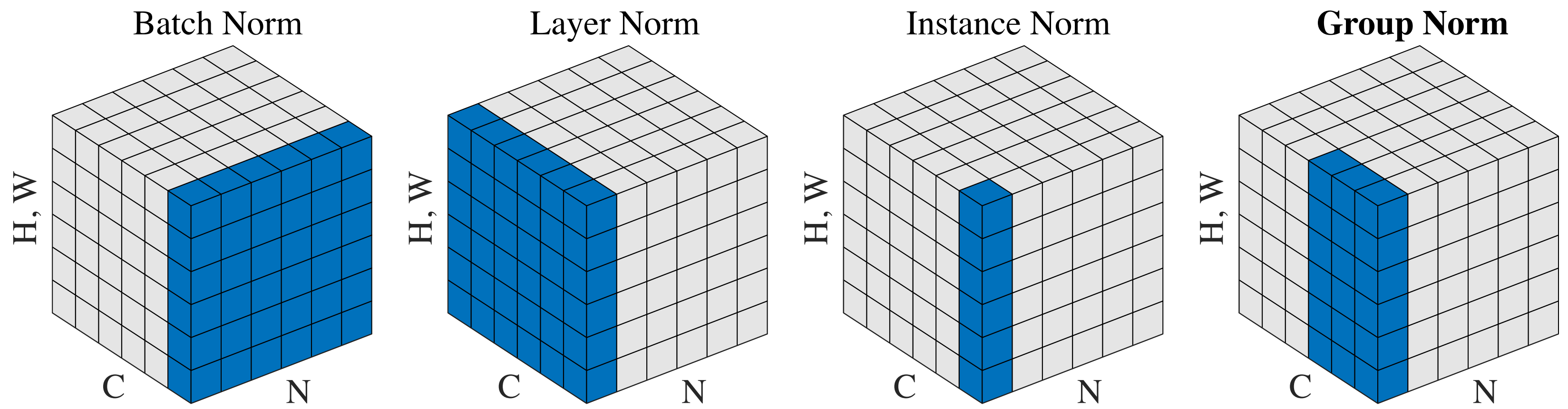
[Optional]
Normalization
across data/features

Output
Features

Normalization

- Lots of different normalization approaches
  - https://mlexplained.com/2018/11/30/an-overview-of-normalization-methods-in-deep-learning/

- Basic idea:
  - Make mean = 0
  - Make standard deviation = 1
  - Question: which dimensions?



https://arxiv.org/pdf/1803.08494.pdf

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: $\mathcal{B} = \{x_{1...m}\}$;
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.

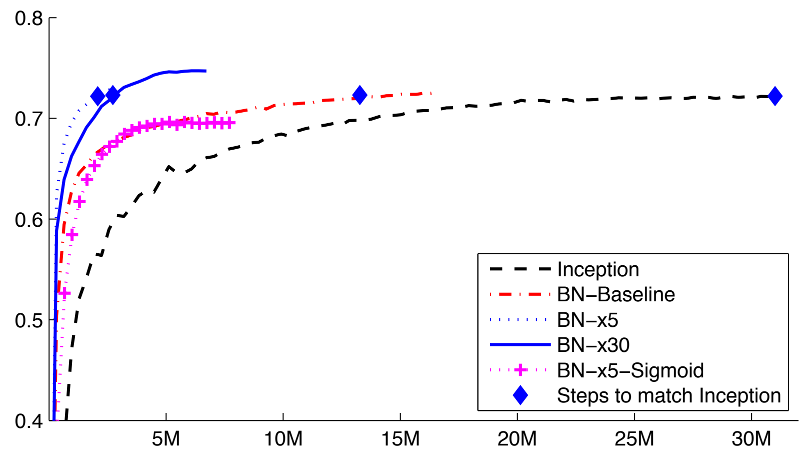
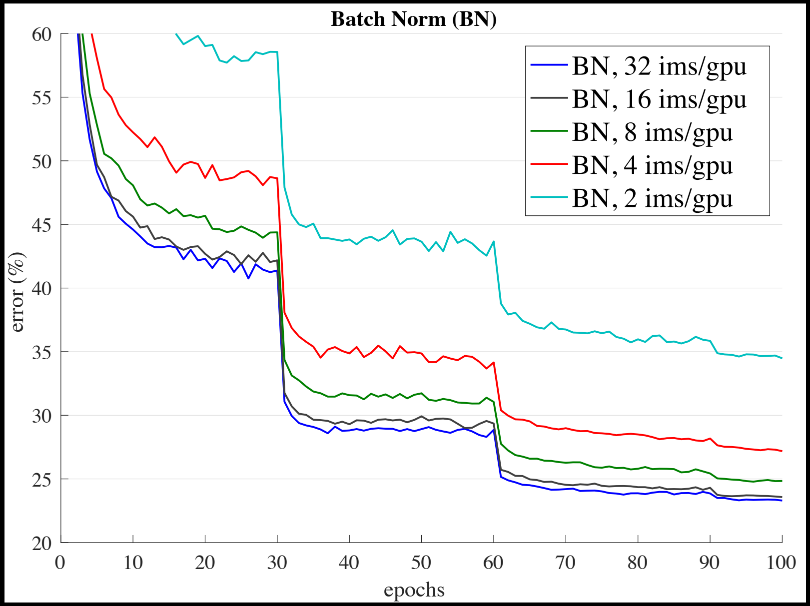


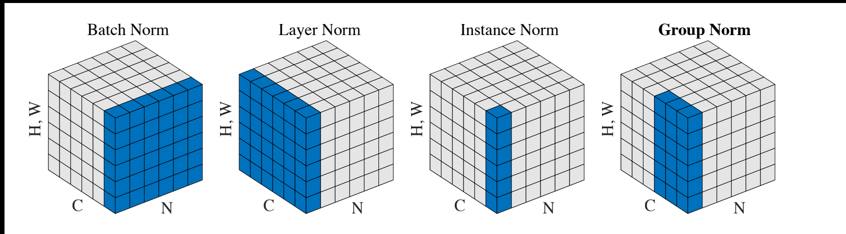
Figure 2: *Single crop validation accuracy of Inception and its batch-normalized variants, vs. the number of training steps.*

Normalization

- But batch normalization has issues, e.g. when batch size is small or 1

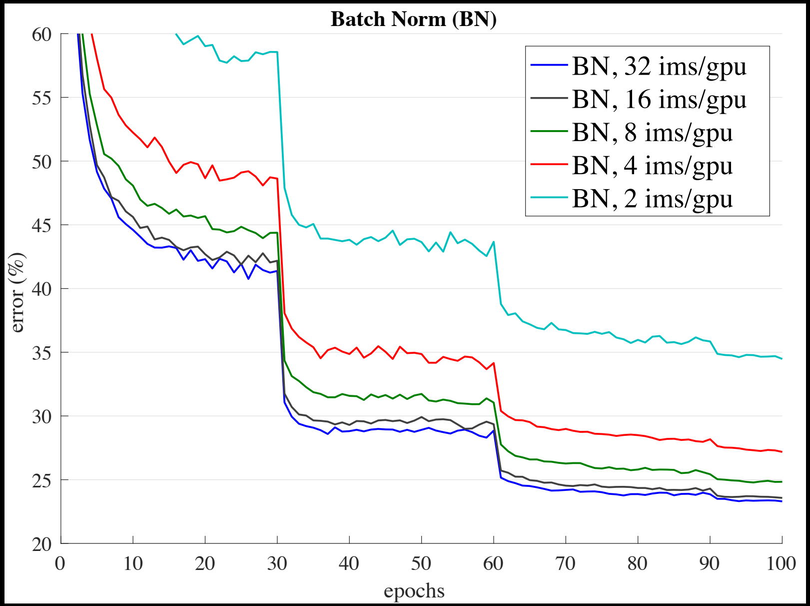


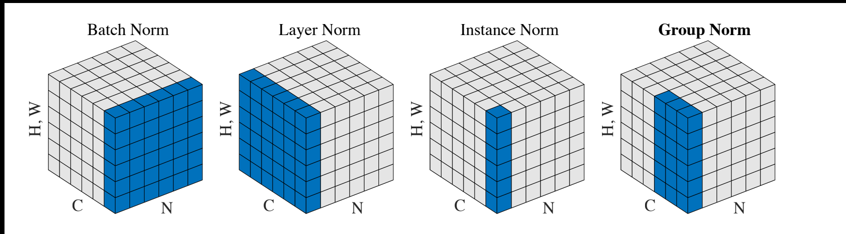
https://arxiv.org/pdf/1803.08494.pdf



Normalization

- But batch normalization has issues, e.g. when batch size is small or 1

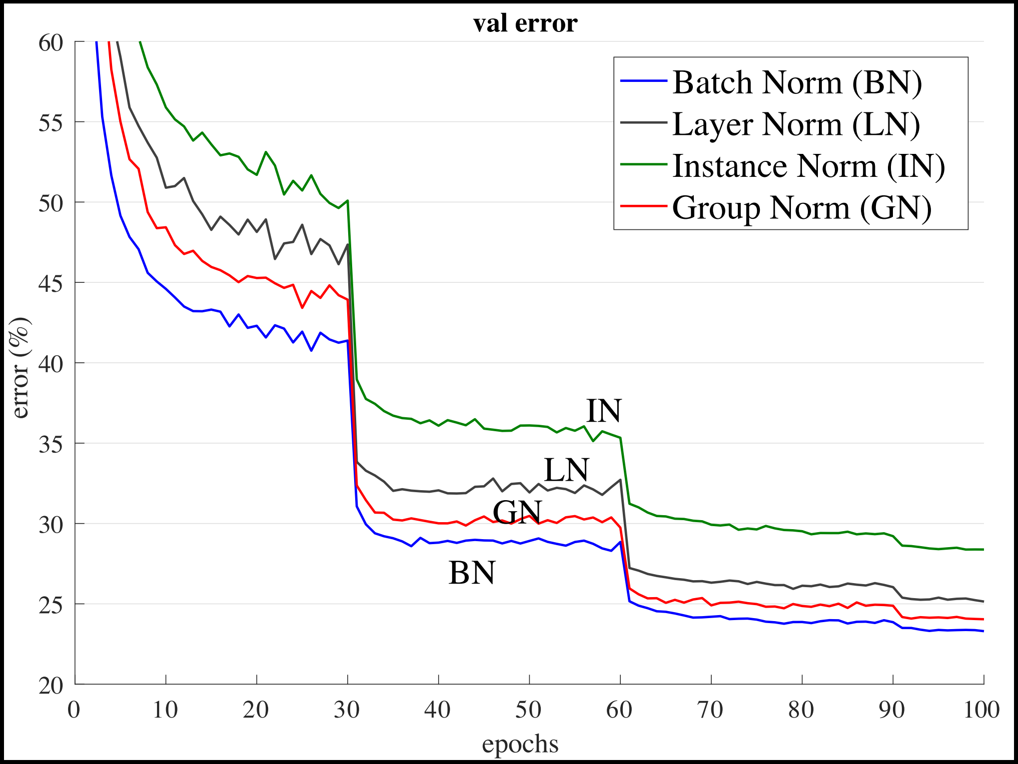


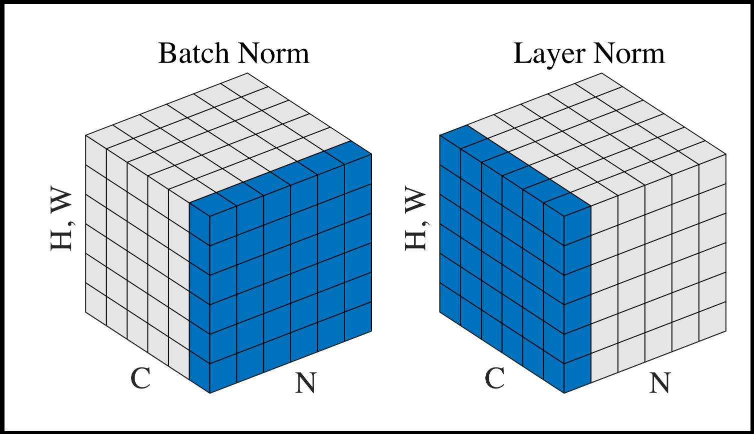


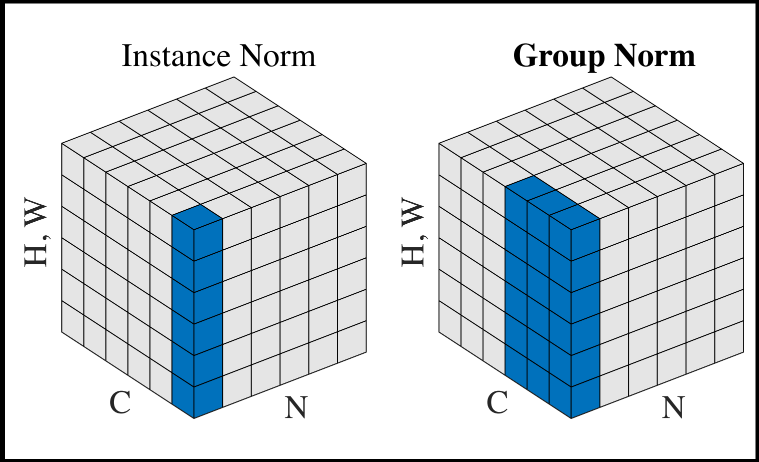
https://arxiv.org/pdf/1803.08494.pdf

Normalization

- Instance Norm, Layer Norm, Group Norm



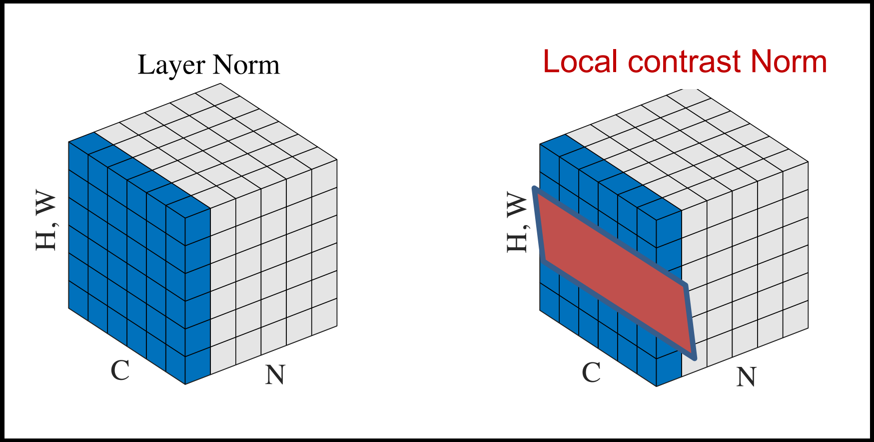




https://arxiv.org/pdf/1803.08494.pdf

Normalization

- Local contrast normalization across features
  - See Divisive Normalization in Neuroscience
  - Local version of Layer Norm

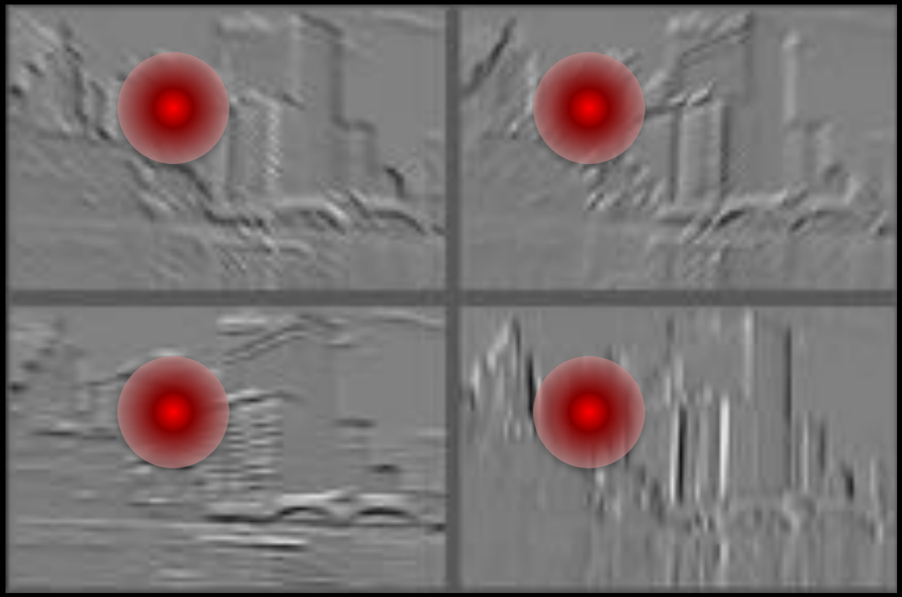


Input

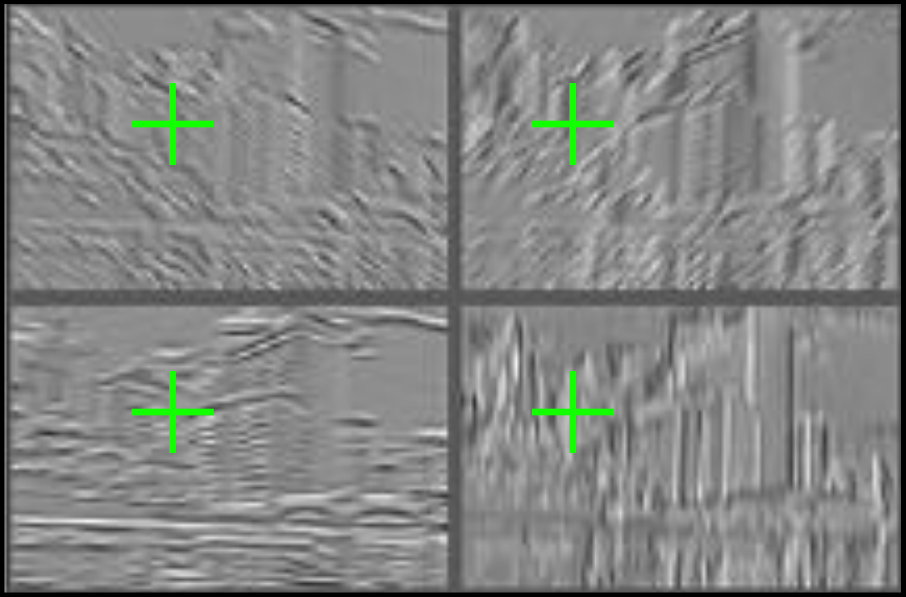
Filters

Normalization

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



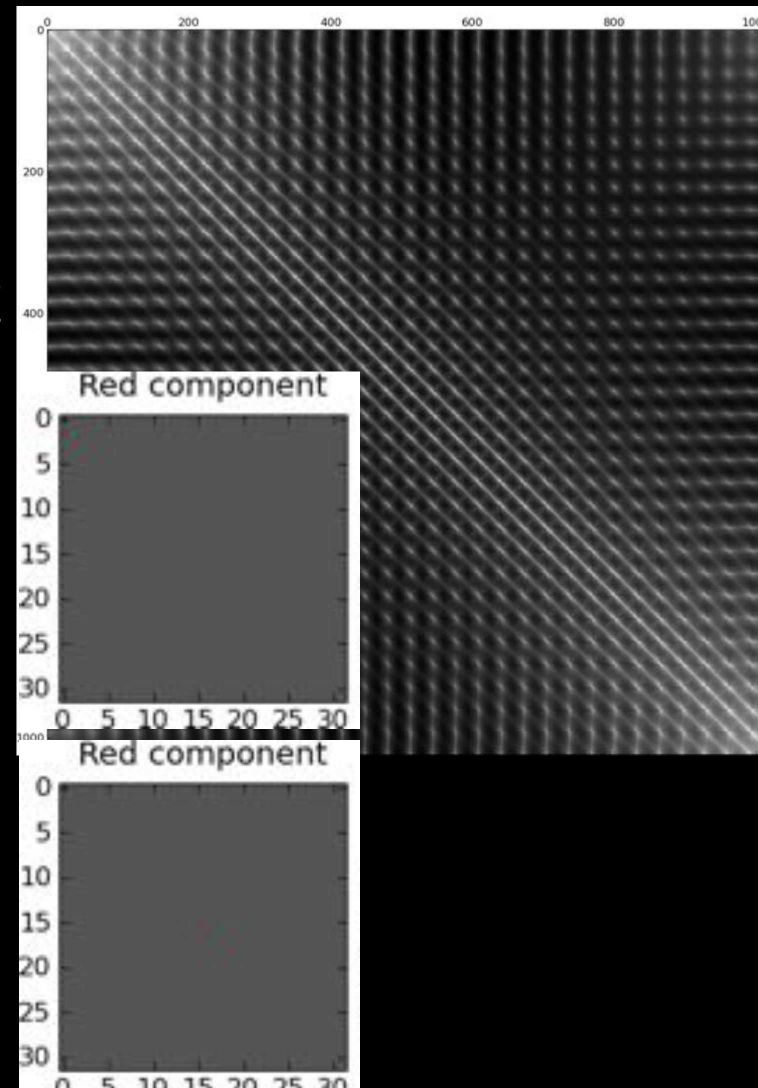
Feature Maps

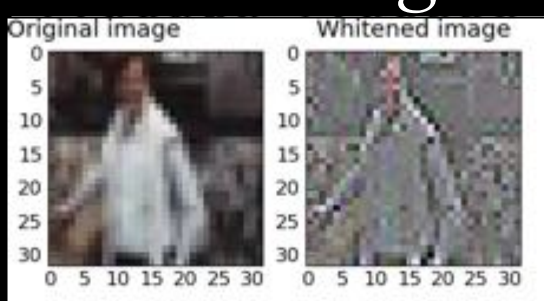


Feature Maps
After Contrast Normalization

Image Whitening

- Convariance matrix of 32x32 real-world images
- Compute whitening matrix W via ZCA transform
- Rows of W, reshaped to 32x32 images
  - Reveals local dependencies
- Whitened image





s.toronto.edu/~kriz/learning-features-2009-TR.pdf

ZCA Transform

- Convariance matrix $C = 1/(n-1)\ X\ X^T$
- Want linear transform W: $Y = W\ X$ such that $YY^T = (n-1)\ I$
- [Some math] $W = (1/(n-1)\ (XX^T))^{-1/2}$
- Compute W using SVD

- Note: only applicable to small images
- For large images, use local contrast normalization

https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

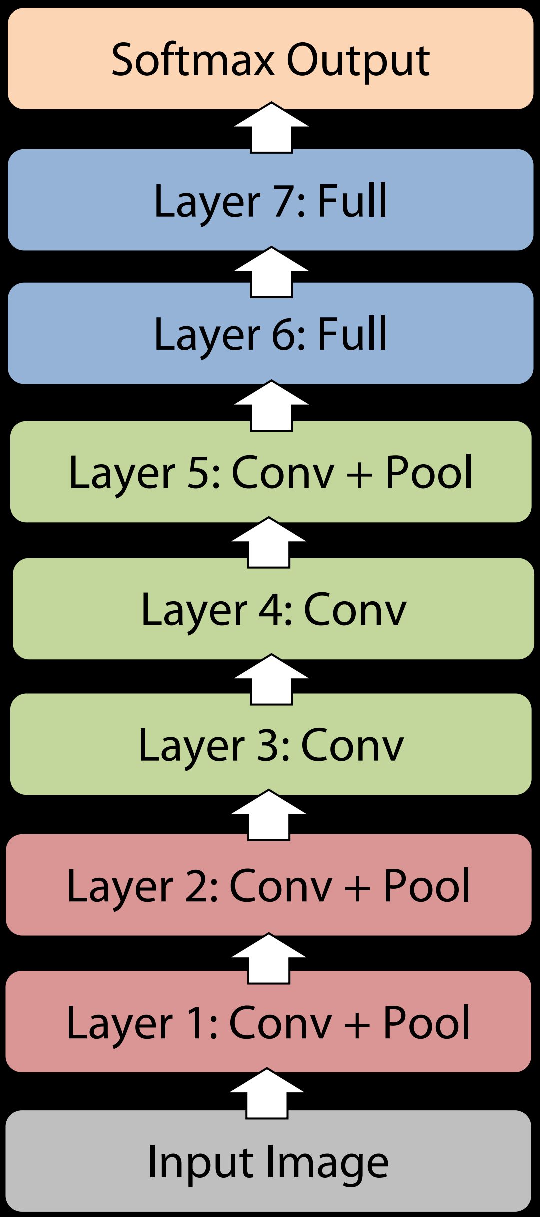
How important is Depth

- "Deep" in Deep Learning
- Ablation study
- Tap off features

Architecture of Krizhevsky et al.

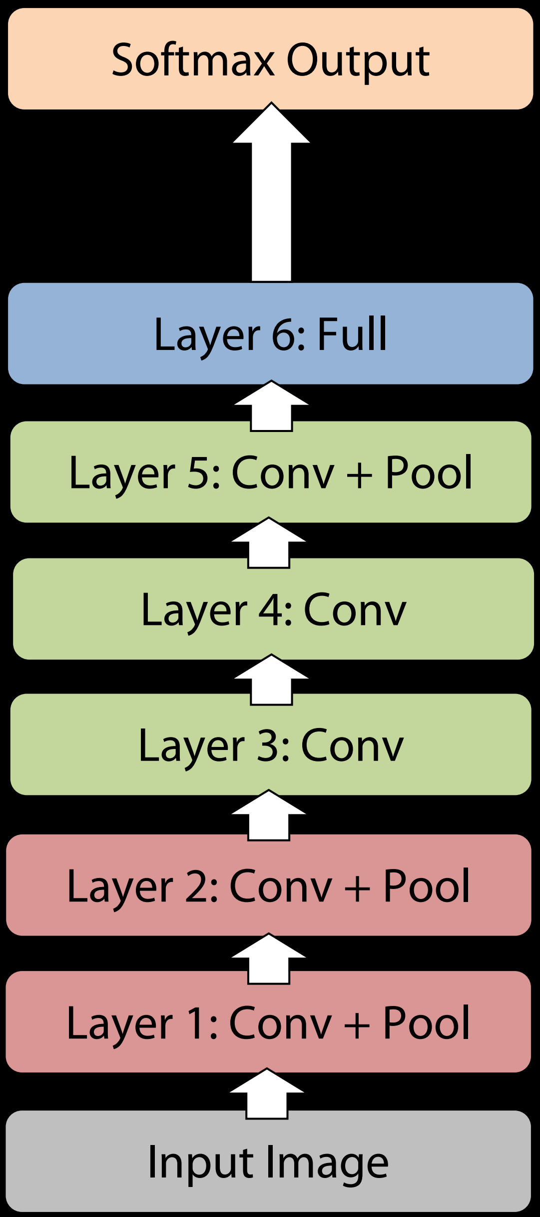
- 8 layers total
- Trained on Imagenet dataset [Deng et al. CVPR'09]
- 18.2% top-5 error
- Our reimplementation:

  18.1% top-5 error



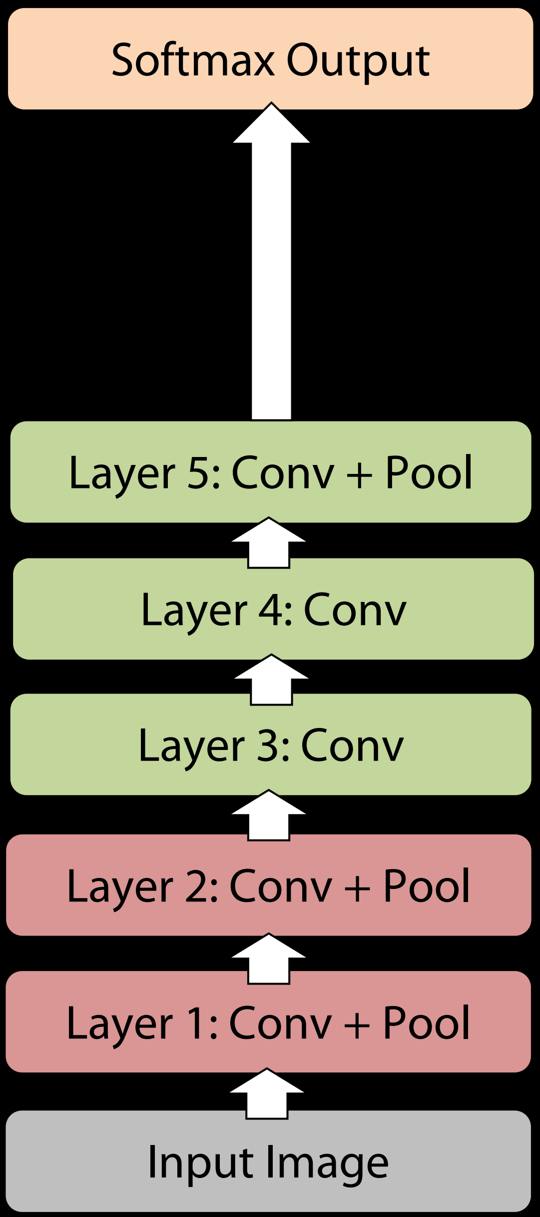
Architecture of Krizhevsky et al.

- 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

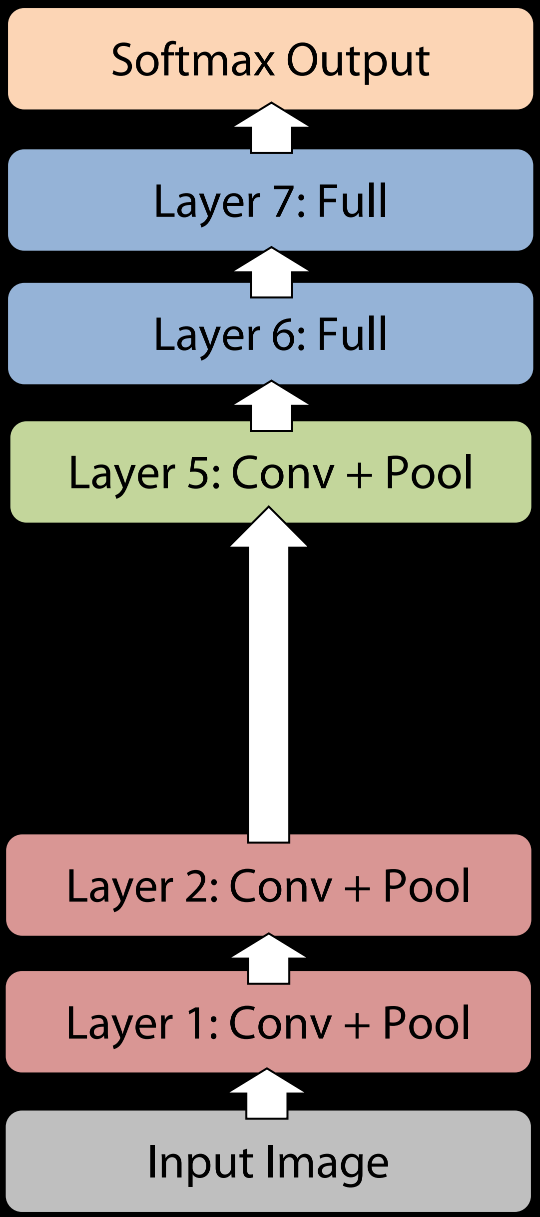


Architecture of Krizhevsky et al.

- Now try removing upper feature extractor layers:
  - Layers 3 & 4

- Drop ~1 million parameters

- 3.0% drop in performance



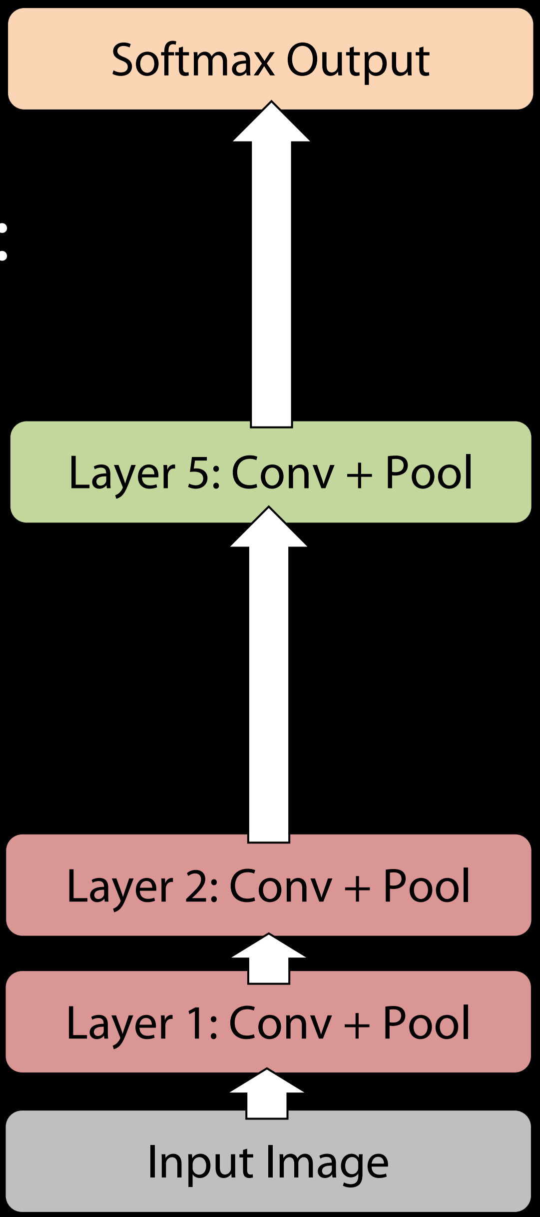
Architecture of Krizhevsky et al.

- 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



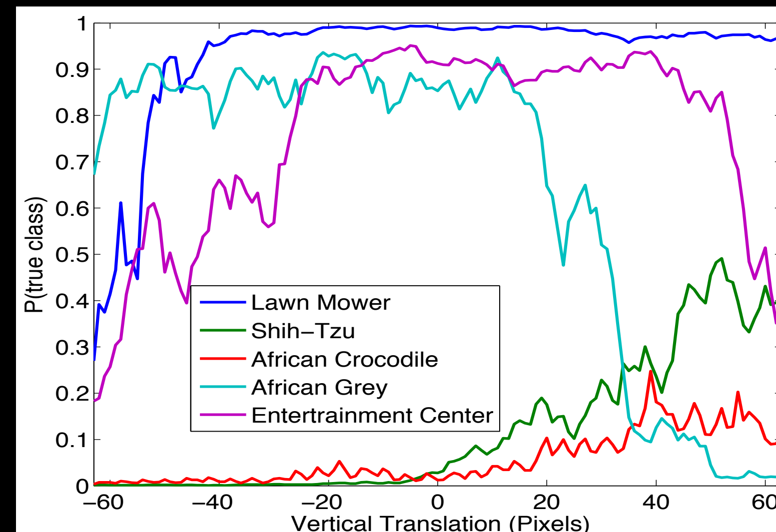
Tapping off Features at each Layer

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

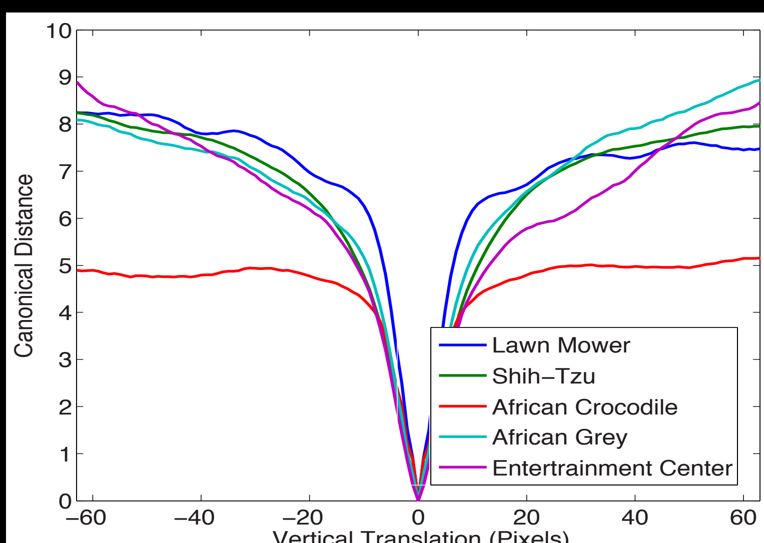
	Cal-101 (30/class)	Cal-256 (60/class)
SVM (1)	44.8 ± 0.7	24.6 ± 0.4
SVM (2)	66.2 ± 0.5	39.6 ± 0.3
SVM (3)	72.3 ± 0.4	46.0 ± 0.3
SVM (4)	76.6 ± 0.4	51.3 ± 0.1
SVM (5)	$\mathbf{86.2 \pm 0.8}$	65.6 ± 0.3
SVM (7)	$\mathbf{85.5 \pm 0.4}$	$\mathbf{71.7 \pm 0.2}$
Softmax (5)	82.9 ± 0.4	65.7 ± 0.5
Softmax (7)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

Translation (Vertical)

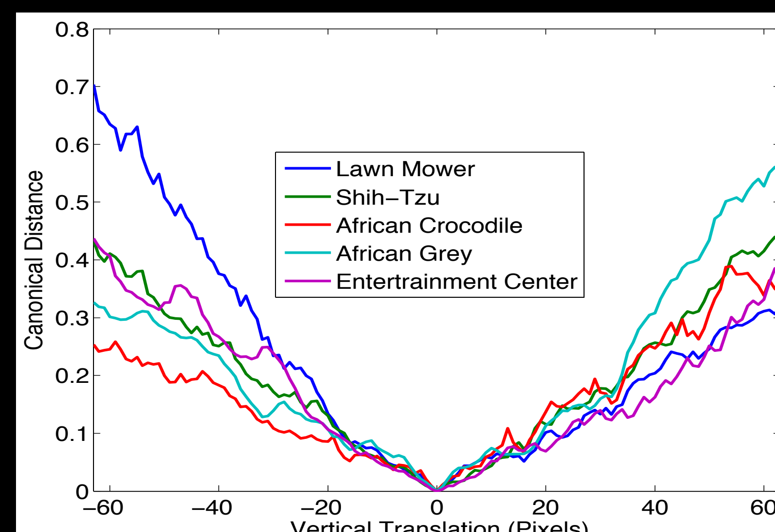
Output



Layer 1

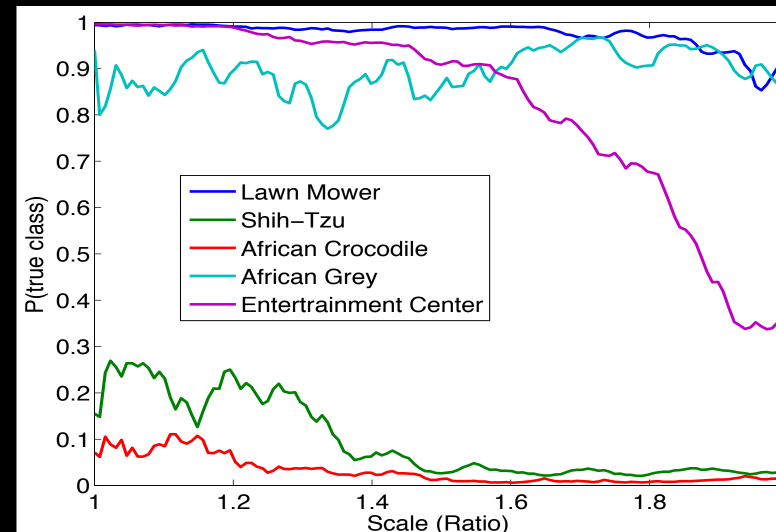


Layer 7

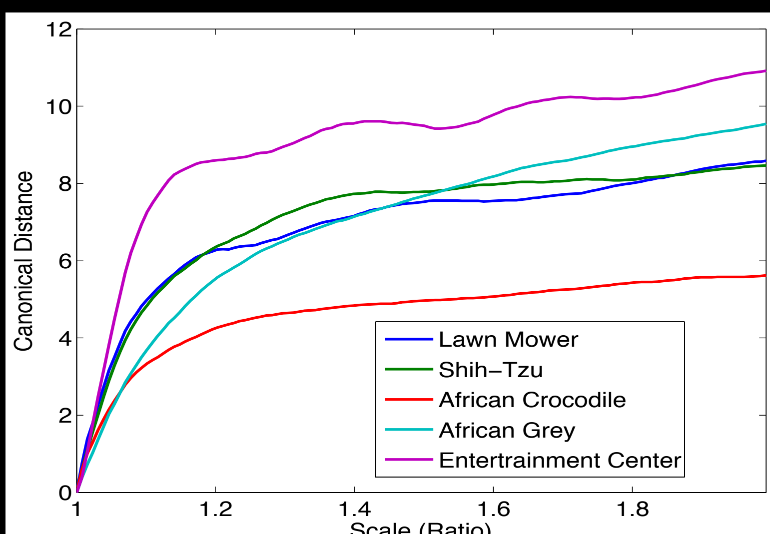


Scale Invariance

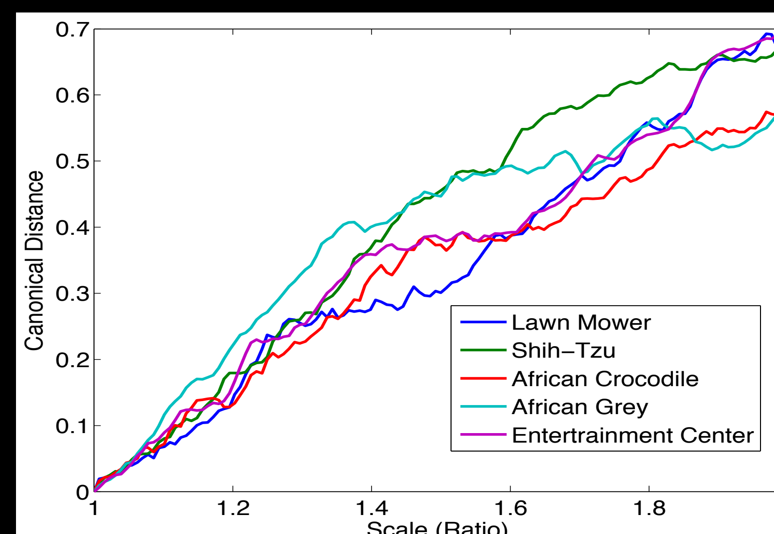
Output



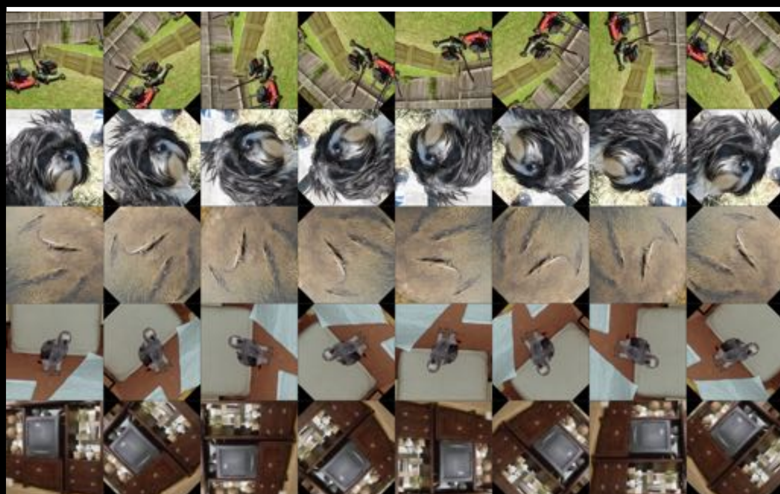
Layer 1



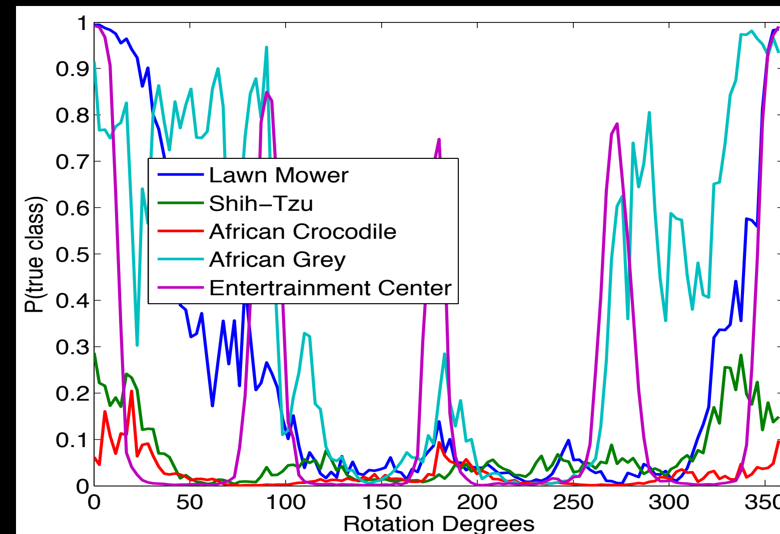
Layer 7



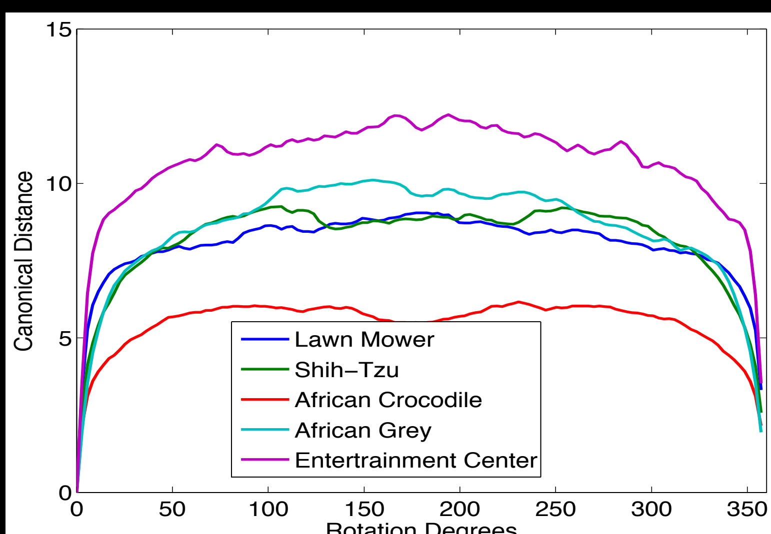
Rotation Invariance



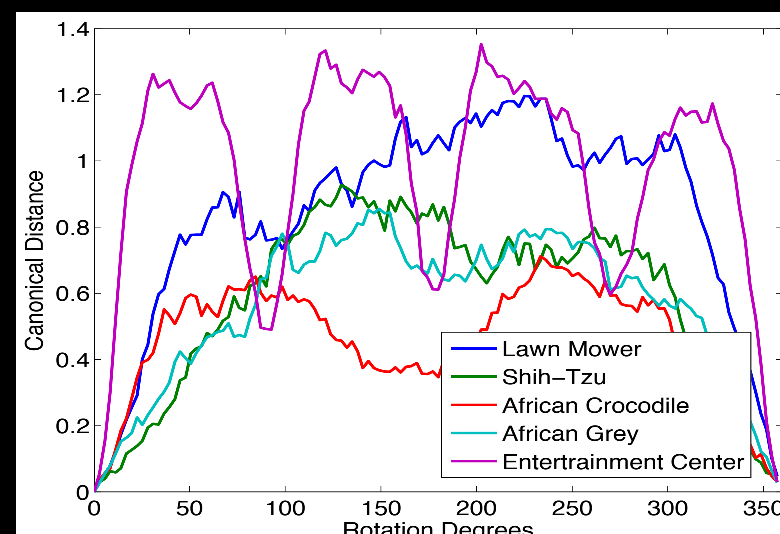
Output



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		top-1 val. error (%)	top-5 val. error (%)
	train (S)	test (Q)		
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**

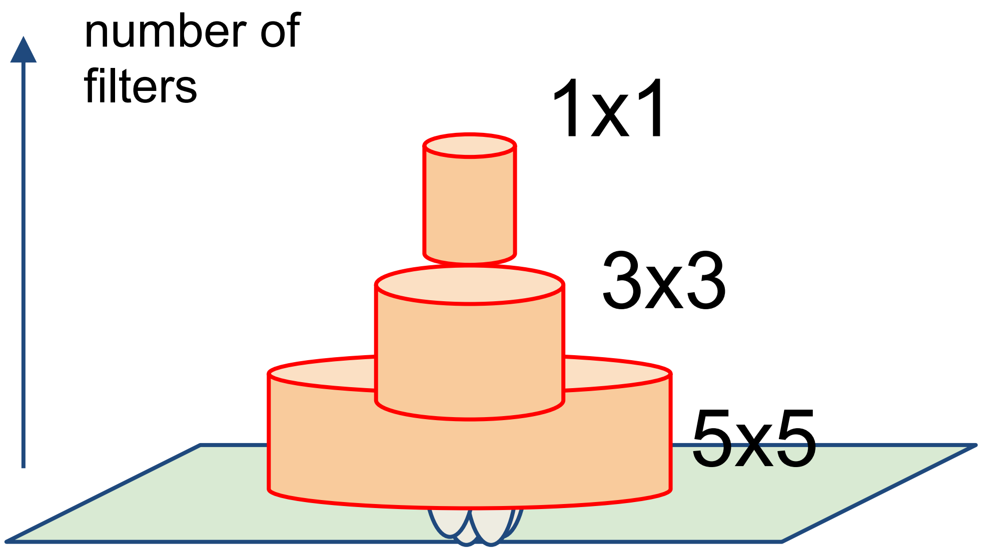
Very Deep Models (2)

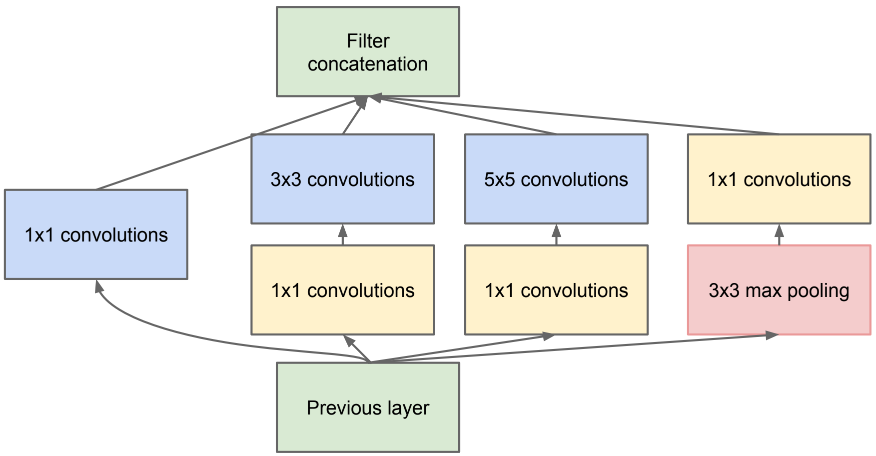
[Going Deep with Convolutions, Szegedy et al., arXiv:1409.4842, 2014]

GoogLeNet inception module:

1. Multiple filter scales at each layer

2. Dimensionality reduction to keep computational requirements down

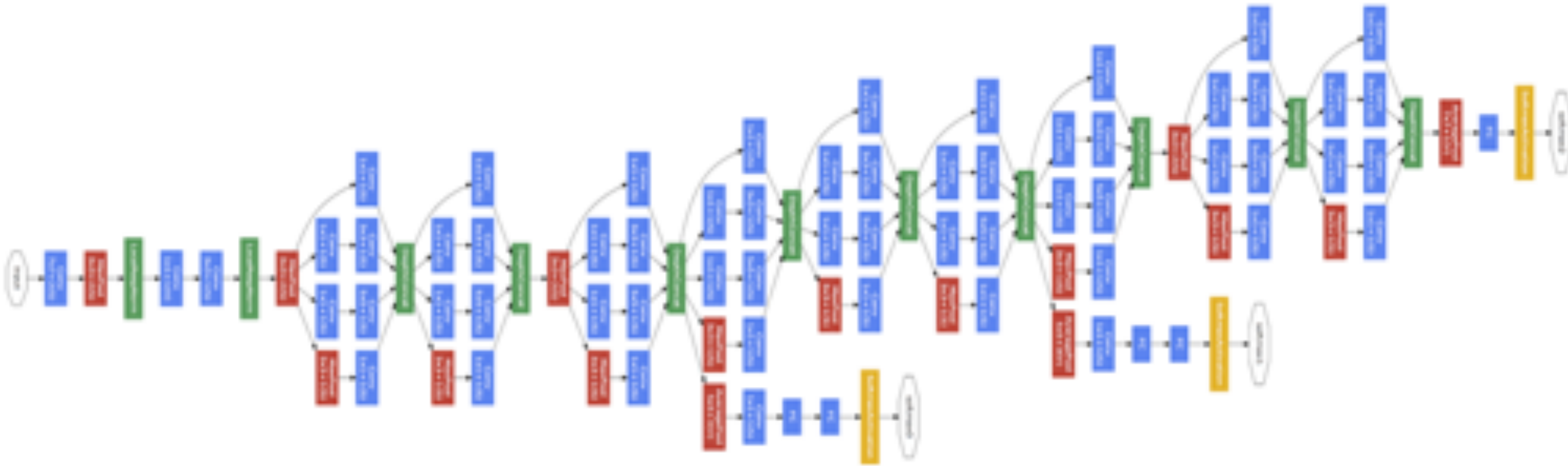




[From http://image-
net.org/challenges/LSVRC/2014/slides/Go

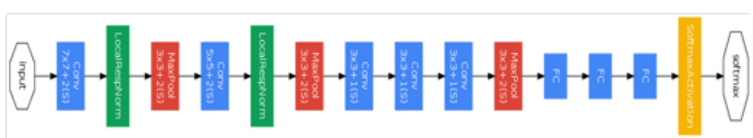
GoogLeNet vs Previous Models

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



GoogLeNet

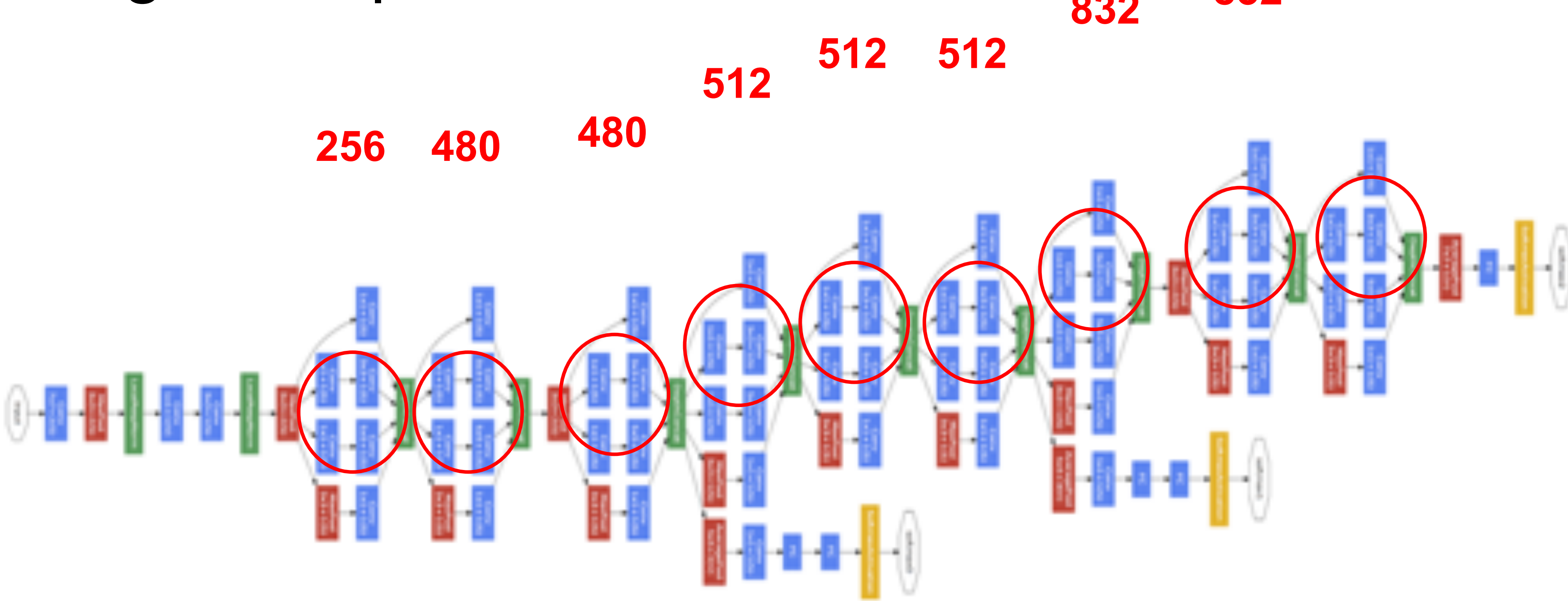
Convolution
Pooling
Softmax
Other



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

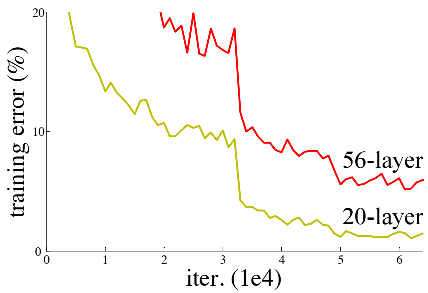
[From http://image-net.org/challenges/LSVRC/2014/slides/Go

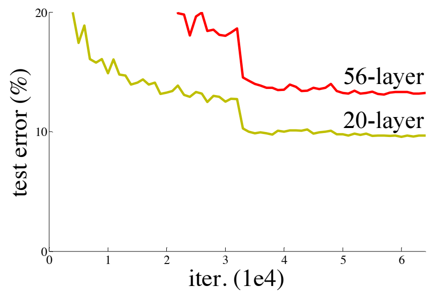
Computional cost is increased by less than 2X compared to Krizhevsky's network. (<1.5Bn operations/evaluation)

Residual Networks

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

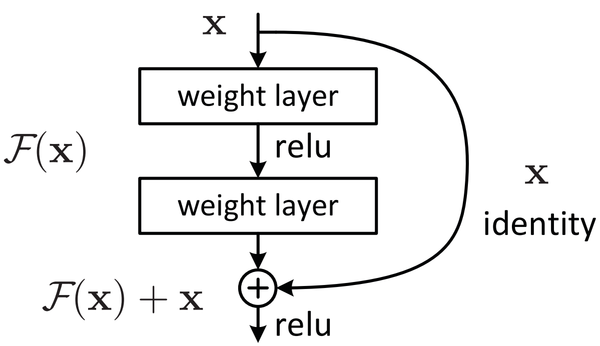
Really, really deep convnets don't train well,
E.g. CIFAR10:





Key idea: introduce "pass through" into each layer

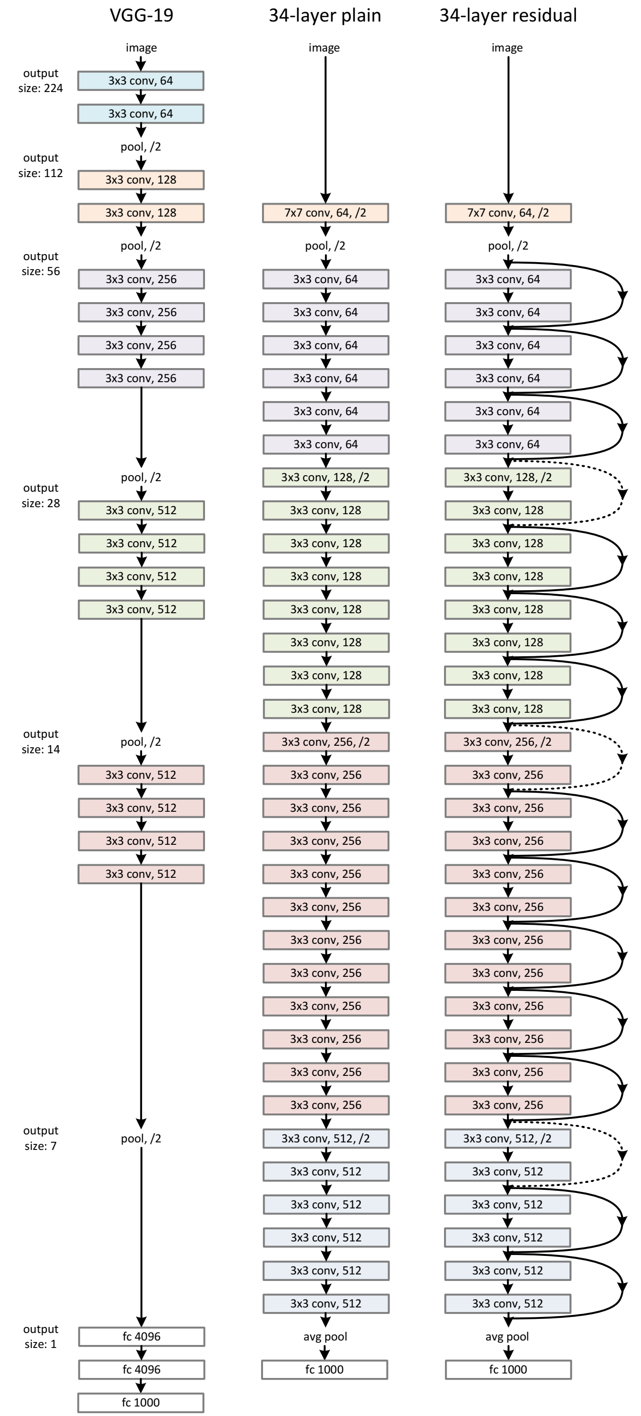
Thus only residual now needs to be learned



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

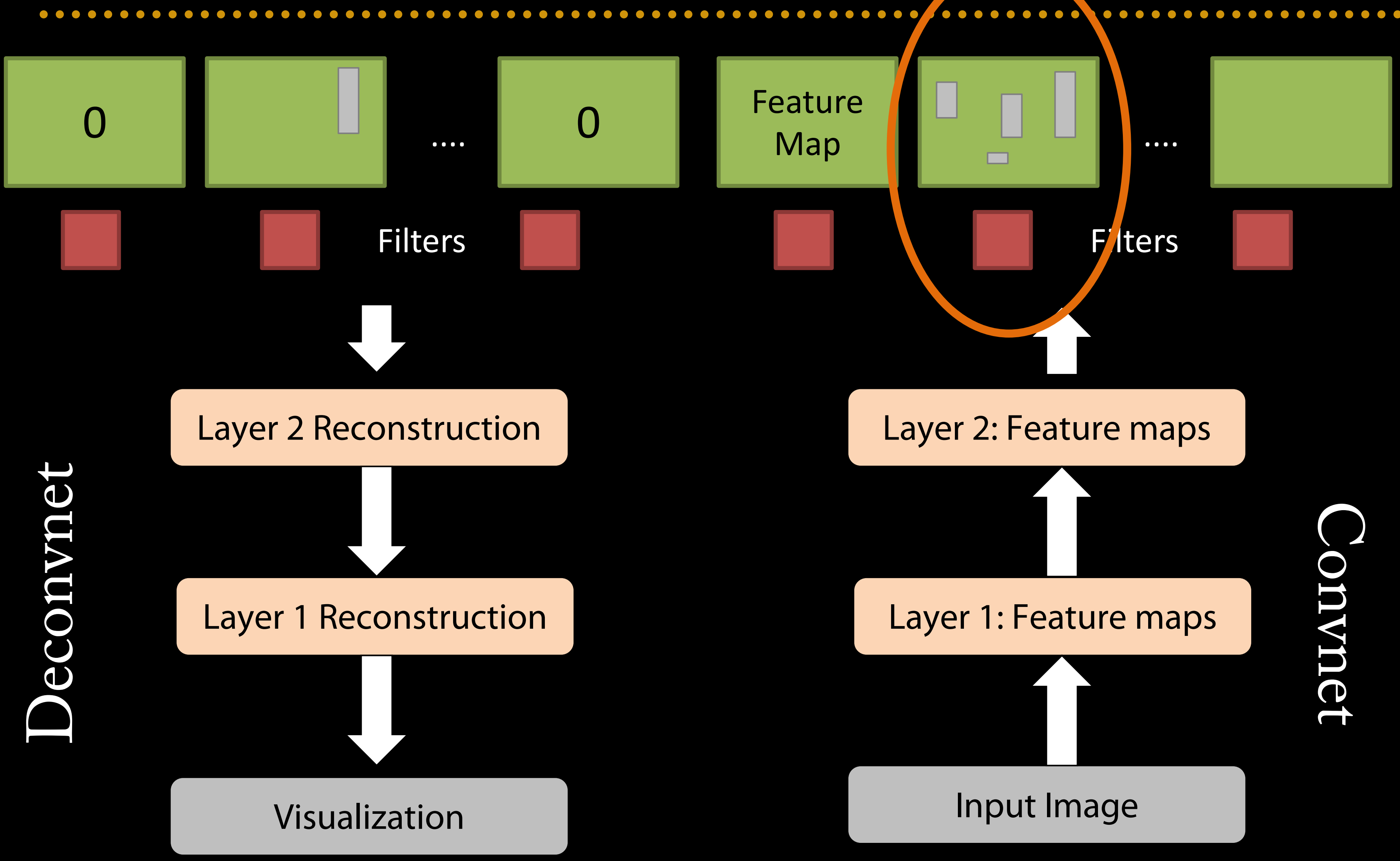
- Want to know what they are learning
- Raw coefficients of learned filters in higher layers difficult to interpret
- Two classes of method:
  1. Project activations back to pixel space
  2. Optimize input image to maximize a particular feature map or class

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

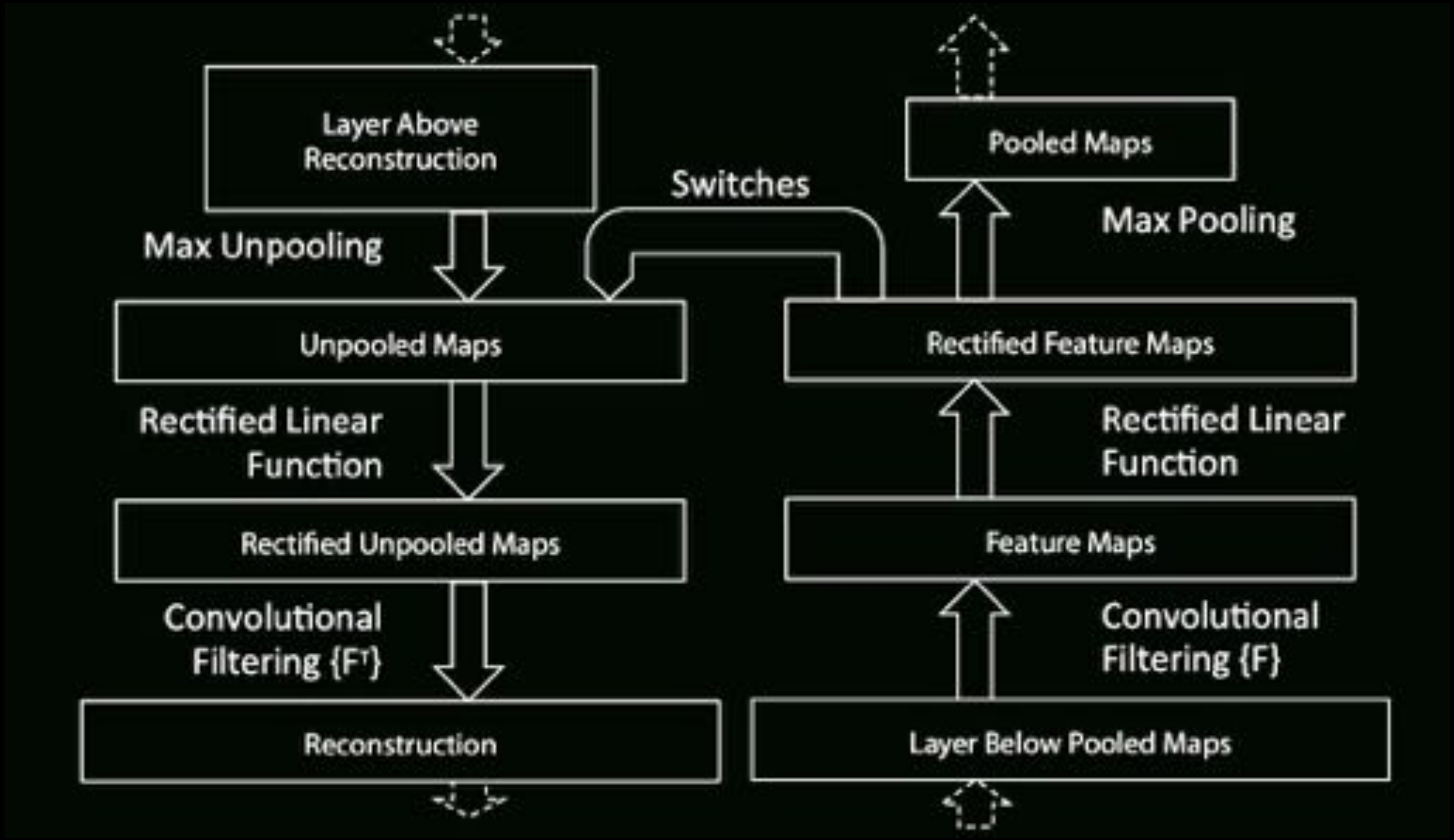
[Zeiler et al. ECCV14]



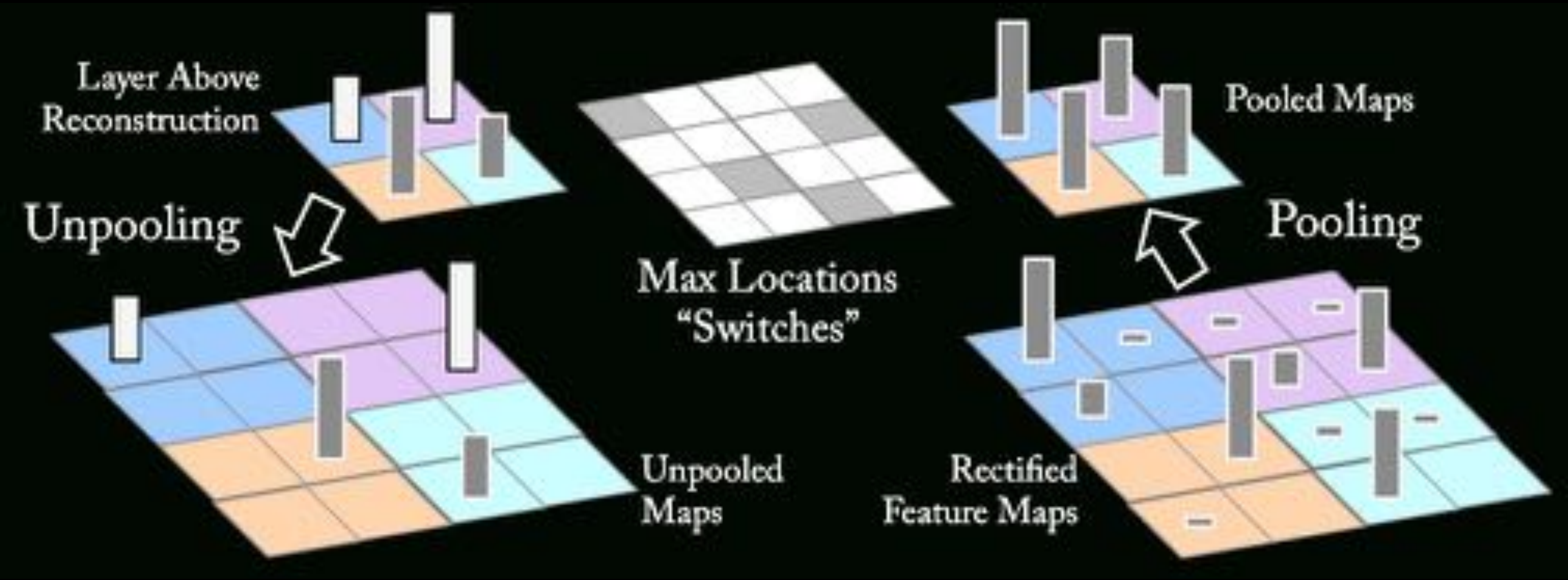
Details of Operation

Deconvnet layer

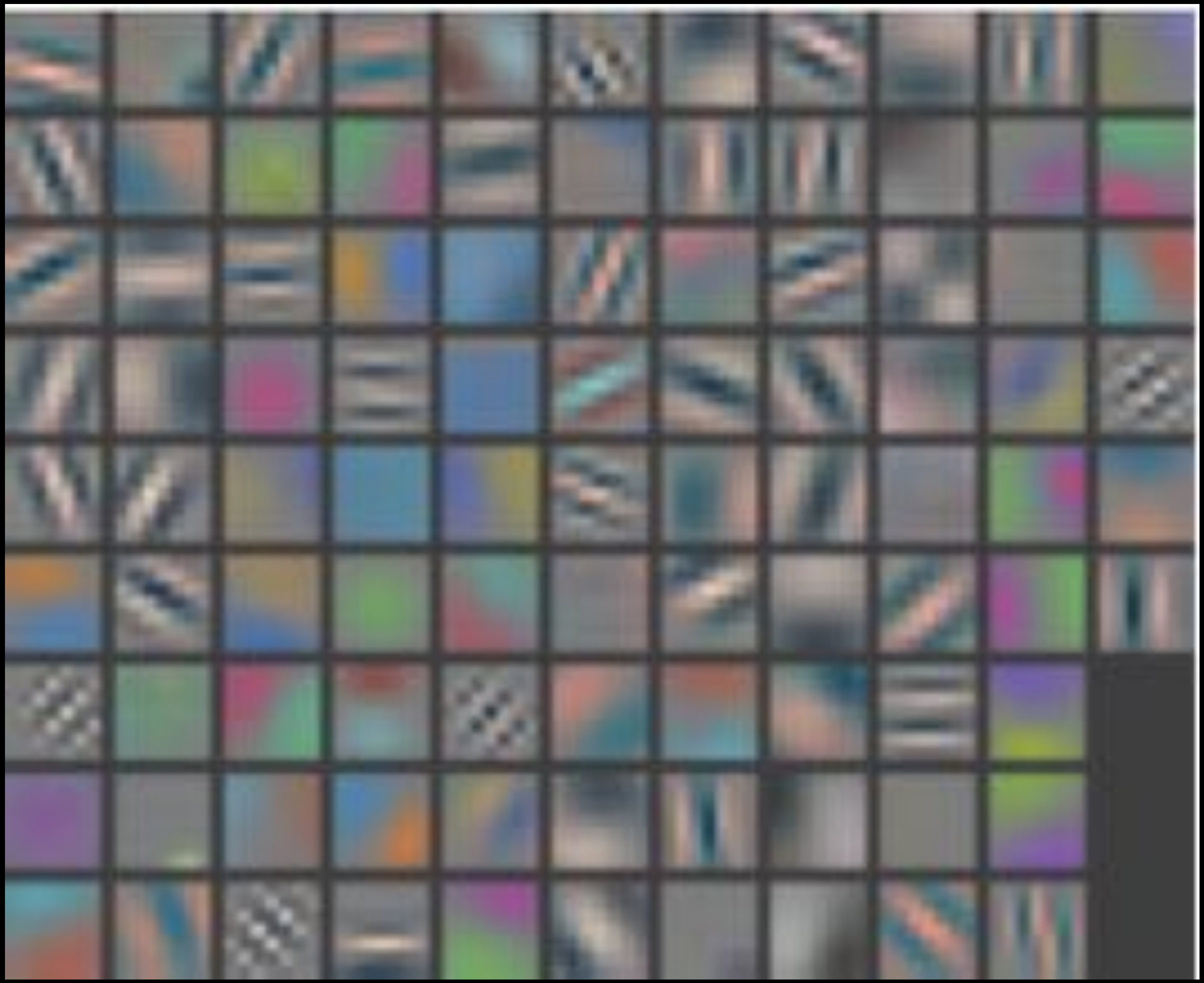
Convnet layer



Unpooling Operation

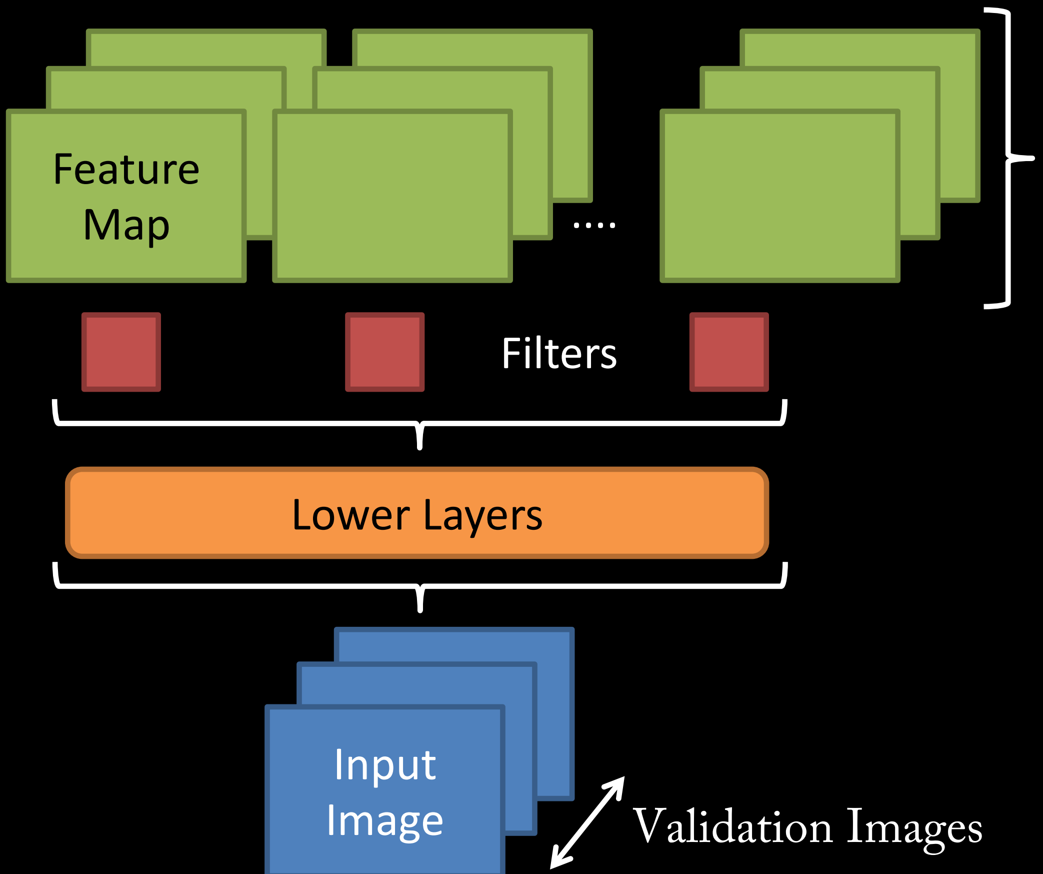


Layer 1 Filters



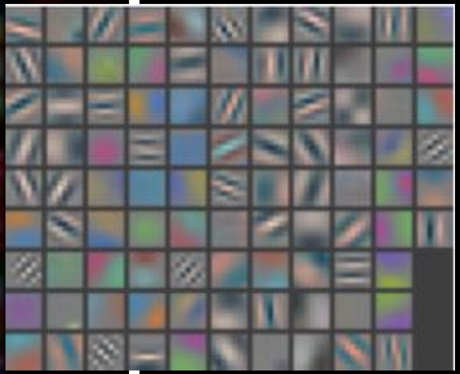
Visualizations of Higher Layers

- 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

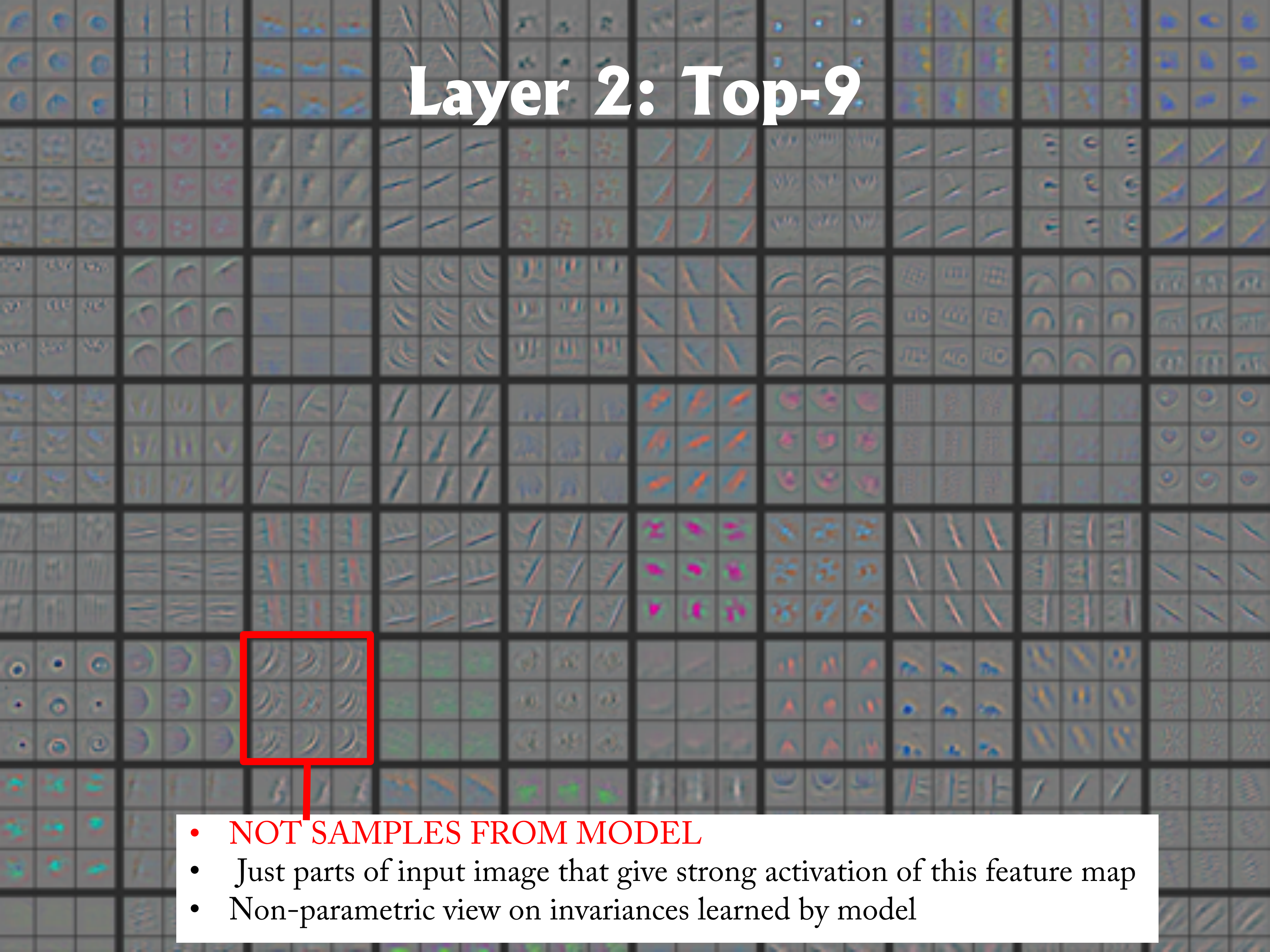
Layer 1: Top-9 Patches



Layer 2: Top-9 Patches

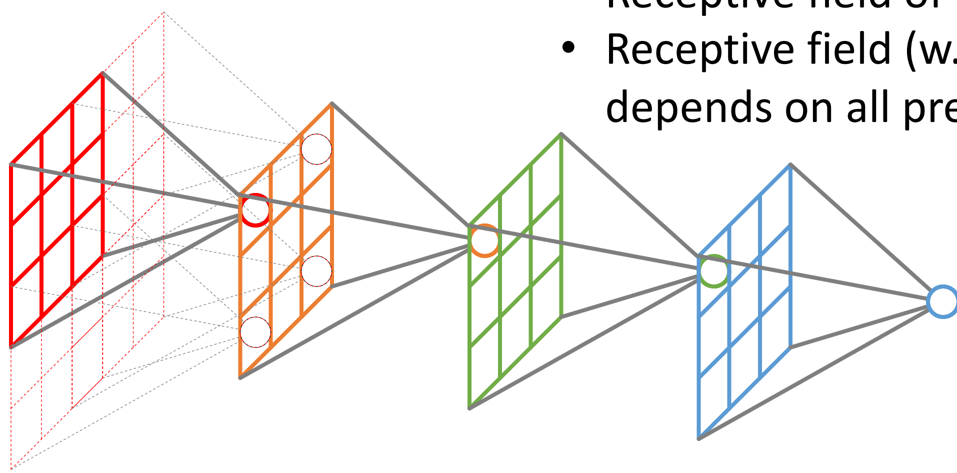
- Patches from validation images that give maximal activation of a given feature map

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

Receptive Field



- Receptive field of the first layer is the filter size
- Receptive field (w.r.t. input image) of a deeper layer depends on all previous layers' filter size and strides

- Correspondence between a feature map pixel and an image pixel is not unique
- Map a feature map pixel to the center of the receptive field on the image in the SPP-net paper

Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition". ECCV 2014.

Layer 3: Top-9

Layer 4: Top-9 Patches

Layer 4: Top-9

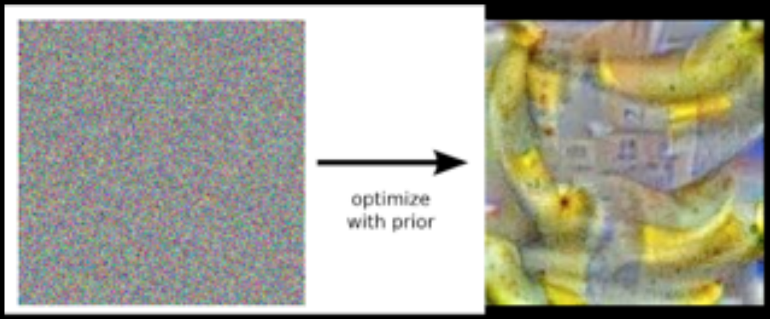
Layer 5: Top-9 Patches

Layer 5: Top-9

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

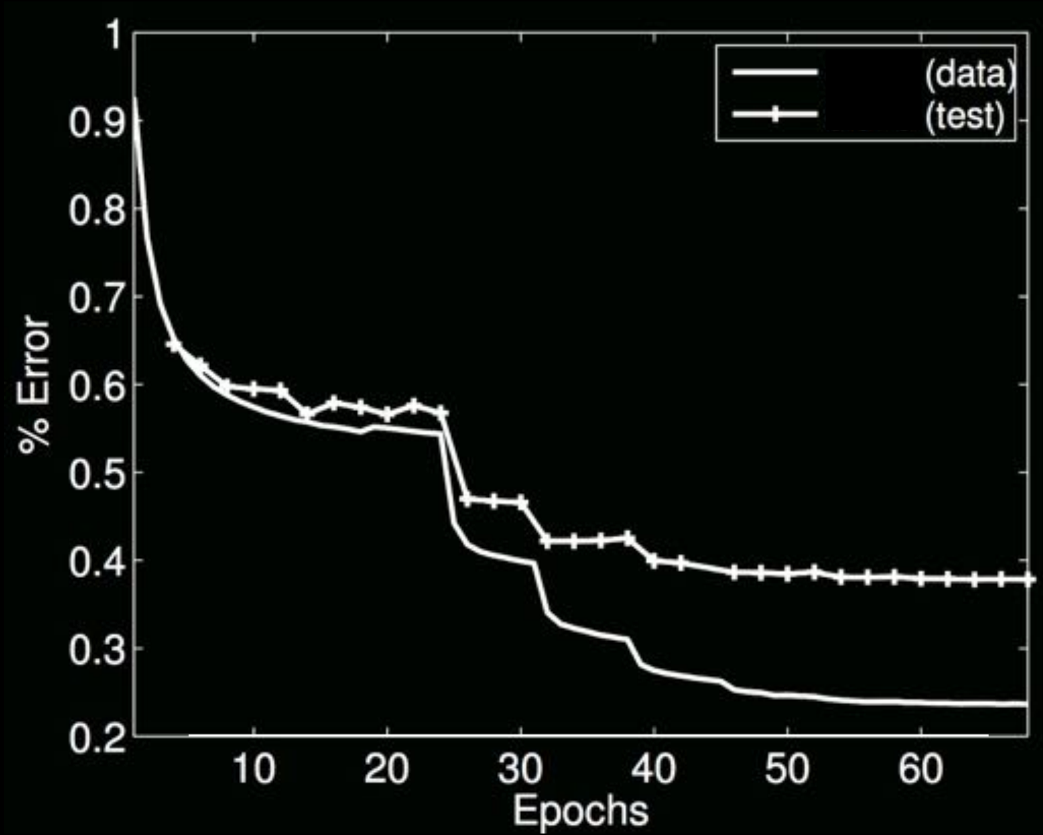
https://photos.google.com/share/F1QipPX0SCl7OzWilt9LnuQliattX4OUCj_8EP65_cTVnBmS1jnYgsGQAieQUc1VQWd
gQ/photo/AF1QipMYTXpt0TvZ0Q5kubkGw8VAq2isxBuL02wKZafB?key=aVBxWjhwSzg2RjJWLWRuVFBBZEN1d20
5bUdEMnhB

Training Big ConvNets

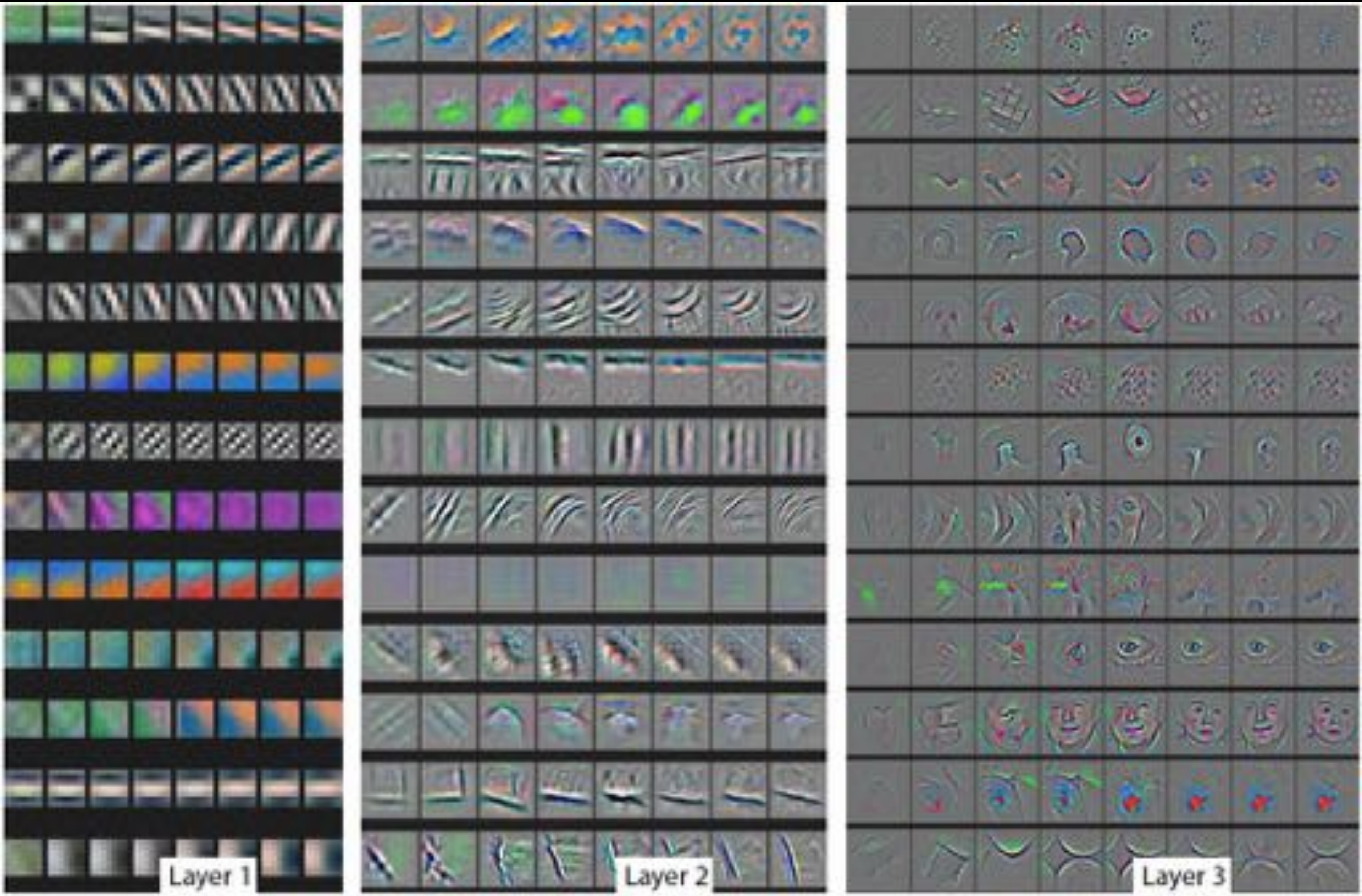
- Stochastic Gradient Descent
  - Compute (noisy estimate of) gradient on small batch of data & make step
  - Take as many steps as possible (even if they are noisy)
  - Large initial learning rate
  - Anneal learning rate
- Momentum
  - Variants [Sutskever ICML 2012]

Annealing of Learning Rate

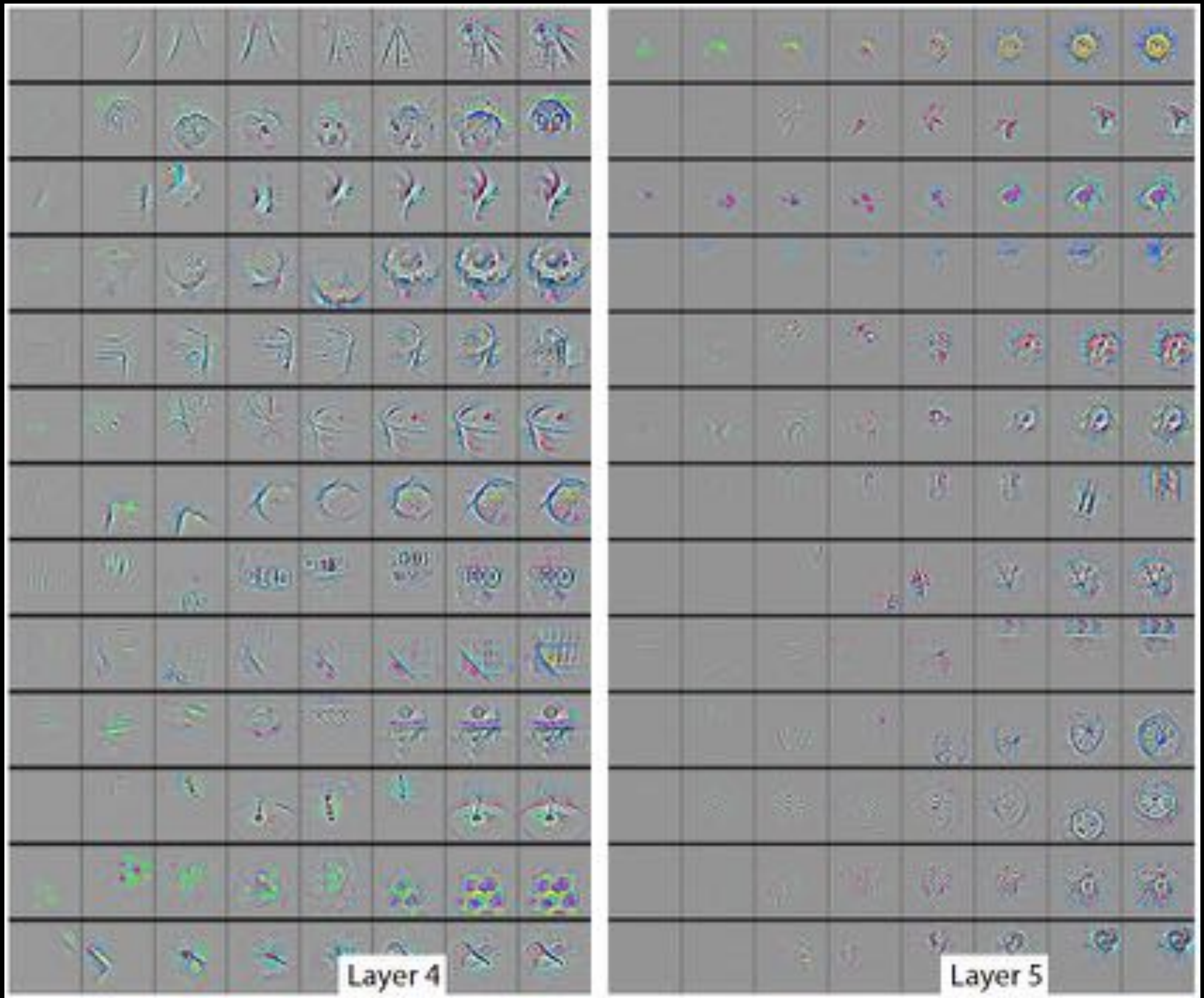
- Start large, slowly reduce
- Explore different scales of energy surface



Evolution of Features During Training



Evolution of Features During Training



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

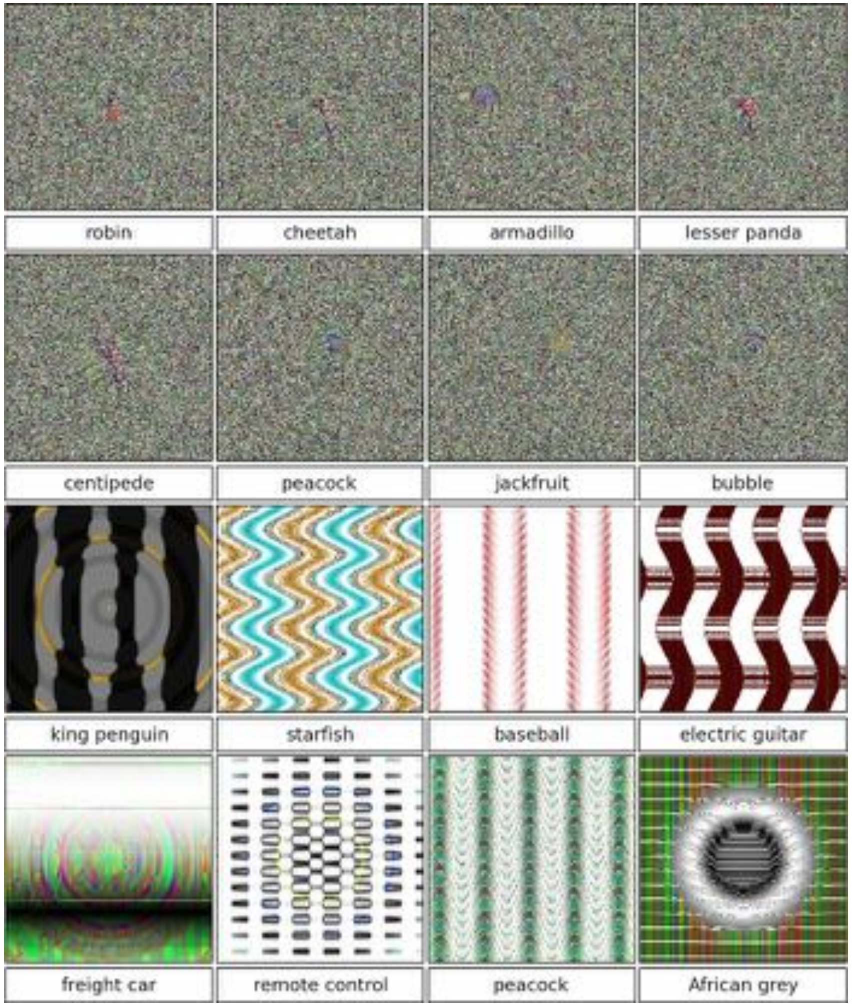
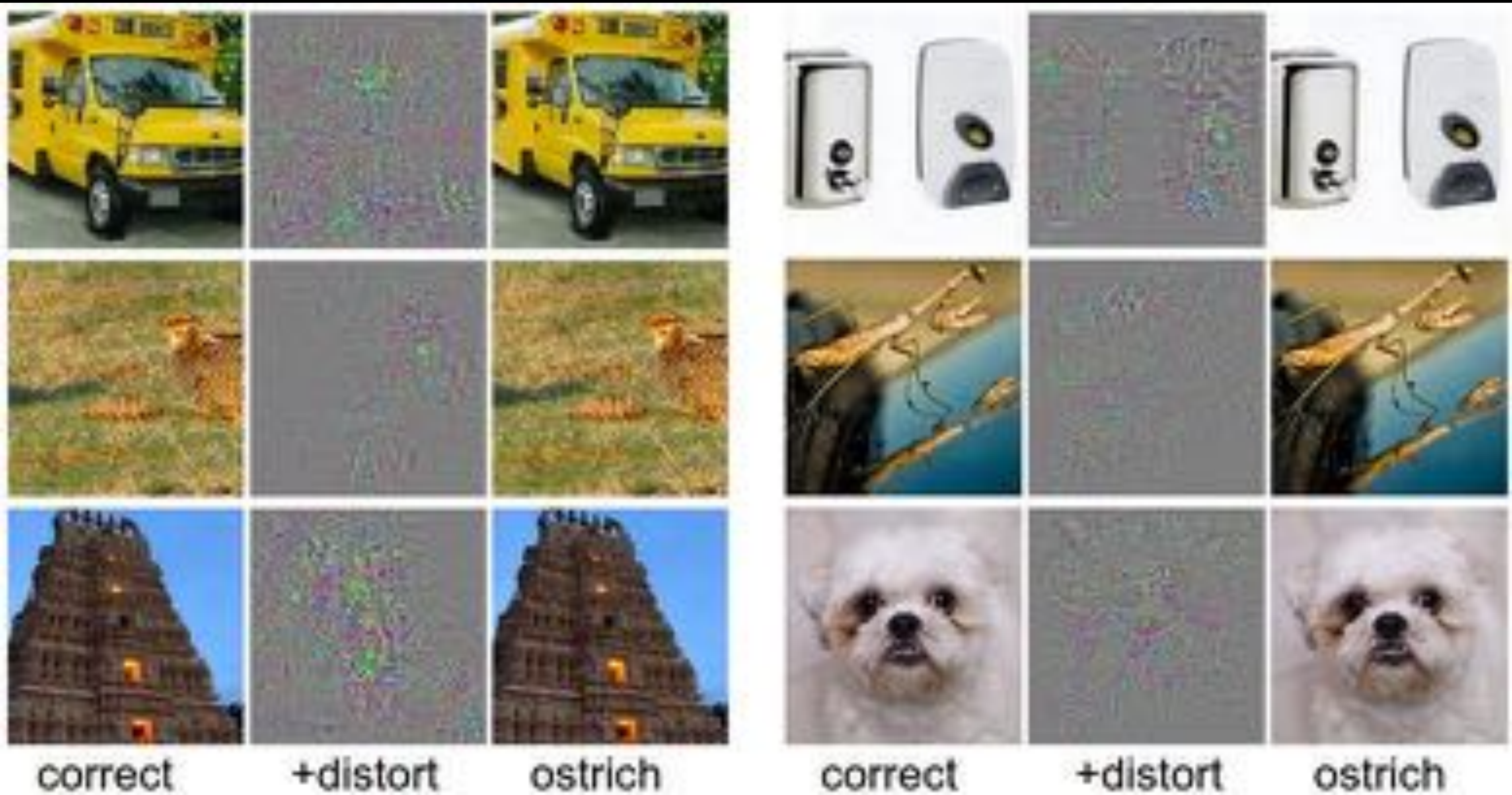


Figure 1. Evolved images that are unrecognizable to humans, but that state-of-the-art DNNs trained on ImageNet believe with $\geq$ 99.6% certainty to be a familiar object. This result highlights differences between how DNNs and humans recognize objects.

Adversarial Examples

- Szegedy et al. arXiv 1312.6199, 2013



Adversarial examples: the formulation

- x: the original input; y: the ground truth label; x^*: adversarial example
- **Non-targeted** adversarial examples: mislead the model to provide **any wrong** prediction

$$\max_{x^*} \ell(f_\theta(x^*), y)$$
$$\text{s.t.} \;\; d(x, x^*) \le B$$

- **Targeted** adversarial examples: mislead the model to provide the **target prediction $\boldsymbol{y^* \neq y}$** specified by the adversary

$$\min_{x^*} \ell(f_\theta(x^*), y^*)$$
$$\text{s.t.} \;\; d(x, x^*) \le B$$

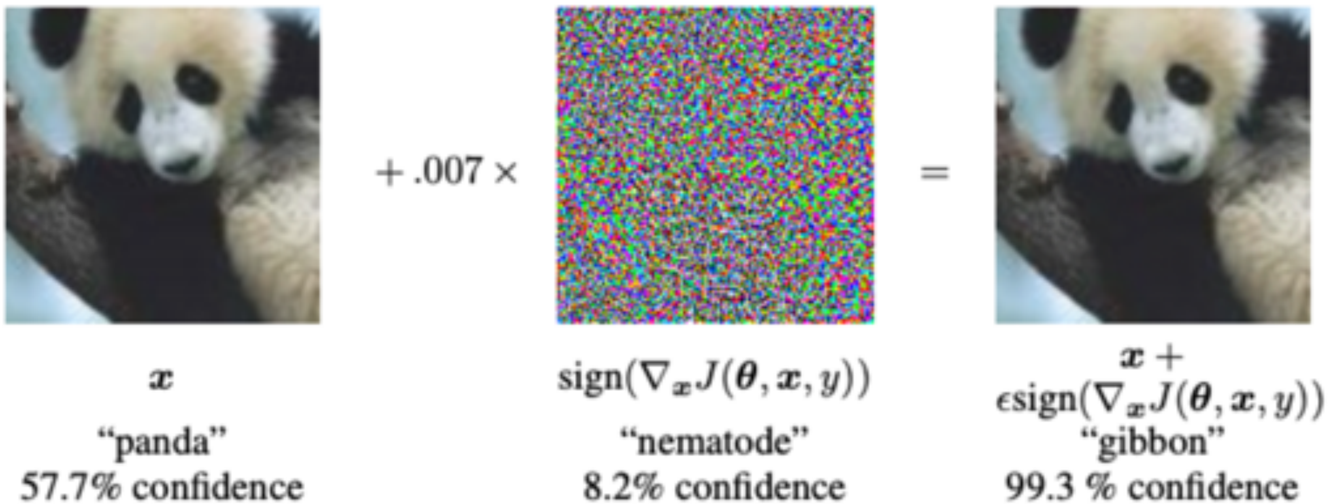
- $d(x, x^*)$ is an ℓ_p norm in most existing work
- B is a constant to make sure that x^* is visually similar to x

Adversarial Attacks in Computer Vision: An Overview, Xinyun Chen, CVPR 2021 tutorial

More material:

- Survey paper: https://arxiv.org/pdf/1911.05268.pdf
- Blog: http://karpathy.github.io/2015/03/30/breaking-convnets/
- CVPR 2021 Tutotal: https://advmlincv.github.io/cvpr21-tutorial/

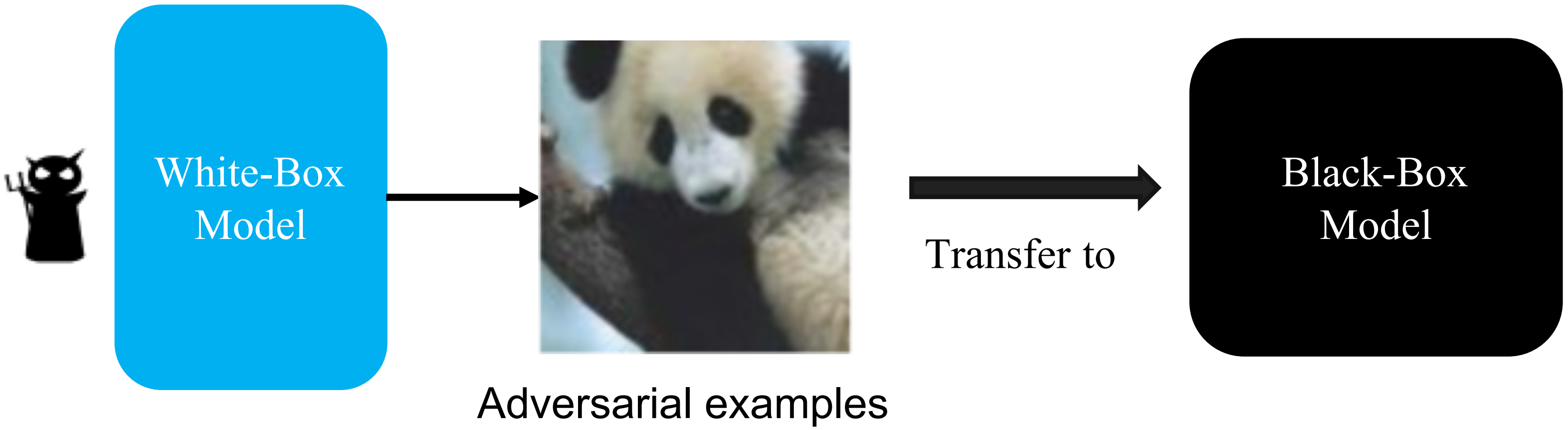
Fast Gradient-Sign Method (FGSM): a one-step attack



- $d(x, x^*)$ is the ℓ_∞ norm
- $x^* = x + B\text{sgn}\left(\nabla_x \ell(f_\theta(x), y)\right)$
- Simple yet effective attacks against models without defense
- Not effective against models with defense

Goodfellow et al. Explaining and Harnessing Adversarial Examples, ICLR 2015.

Black-box attacks based on transferability



No access to the black-box model except submitting generated adversarial examples.

Non-targeted attacks on ImageNet

	RMSD	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
ResNet-152	22.83	0%	13%	18%	19%	11%
ResNet-101	23.81	19%	0%	21%	21%	12%
ResNet-50	22.86	23%	20%	0%	21%	18%
VGG-16	22.51	22%	17%	17%	0%	5%
GoogLeNet	22.58	39%	38%	34%	19%	0%

- RMSD: root mean square deviation $d(x, x^*) = \sqrt{\sum_i (x_i^* - x_i)^2 / M}$, M: image size
- All selected original images are predicted correctly by all models by top-1 accuracy.
- >60% adversarial examples are wrongly classified by different models.

Liu, **Chen**, Liu, Song. Delving into Transferable Adversarial Examples and Black-box Attacks, ICLR 2017.

Transferability of targeted attacks between two models is poor

	ResNet152	ResNet101	ResNet50	VGG16	GoogLeNet	Incept-v3
ResNet152	100%	2%	1%	1%	1%	0%
ResNet101	3%	100%	3%	2%	1%	1%
ResNet50	4%	2%	100%	1%	1%	0%
VGG16	2%	1%	2%	100%	1%	0%
GoogLeNet	1%	1%	0%	1%	100%	0%
Incept-v3	0%	0%	0%	0%	0%	100%

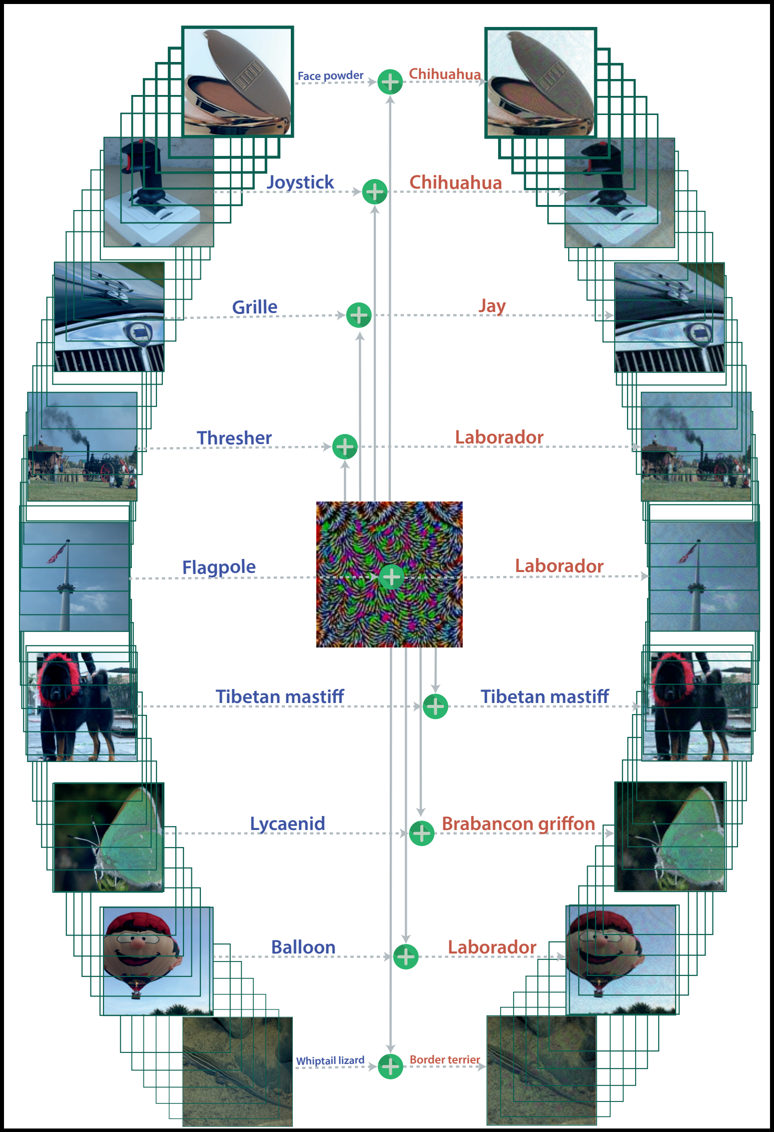
<5% adversarial examples are predicted with the same label by two models.

Ground truth: running shoe

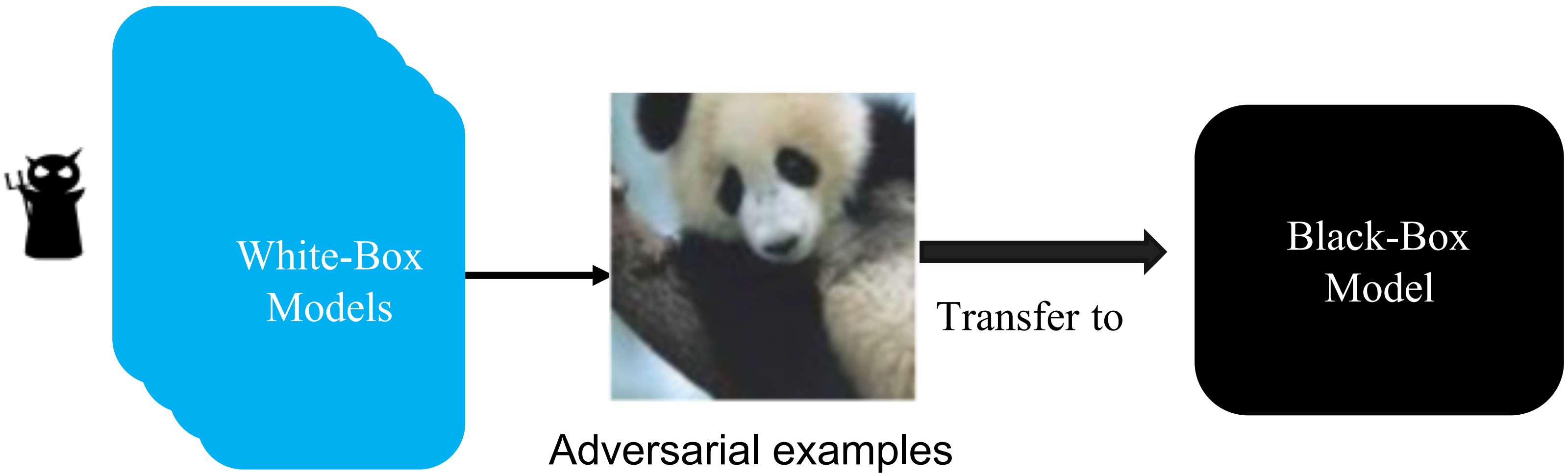
VGG16	**Military uniform**
ResNet50	Jigsaw puzzle
ResNet101	Motor scooter
ResNet152	Mask
GoogLeNet	Chainsaw

Universal Adversarial Examples

- Moosave-Dezfooli et al. arXiv 1610.08401, Oct 2016



Our approach: attacking an ensemble of models



Intuition: If an adversarial example can fool N-1 white-box models, it might transfer better to the N-th black-box model.

Liu, **Chen**, Liu, Song. Delving into Transferable Adversarial Examples and Black-box Attacks, ICLR 2017.

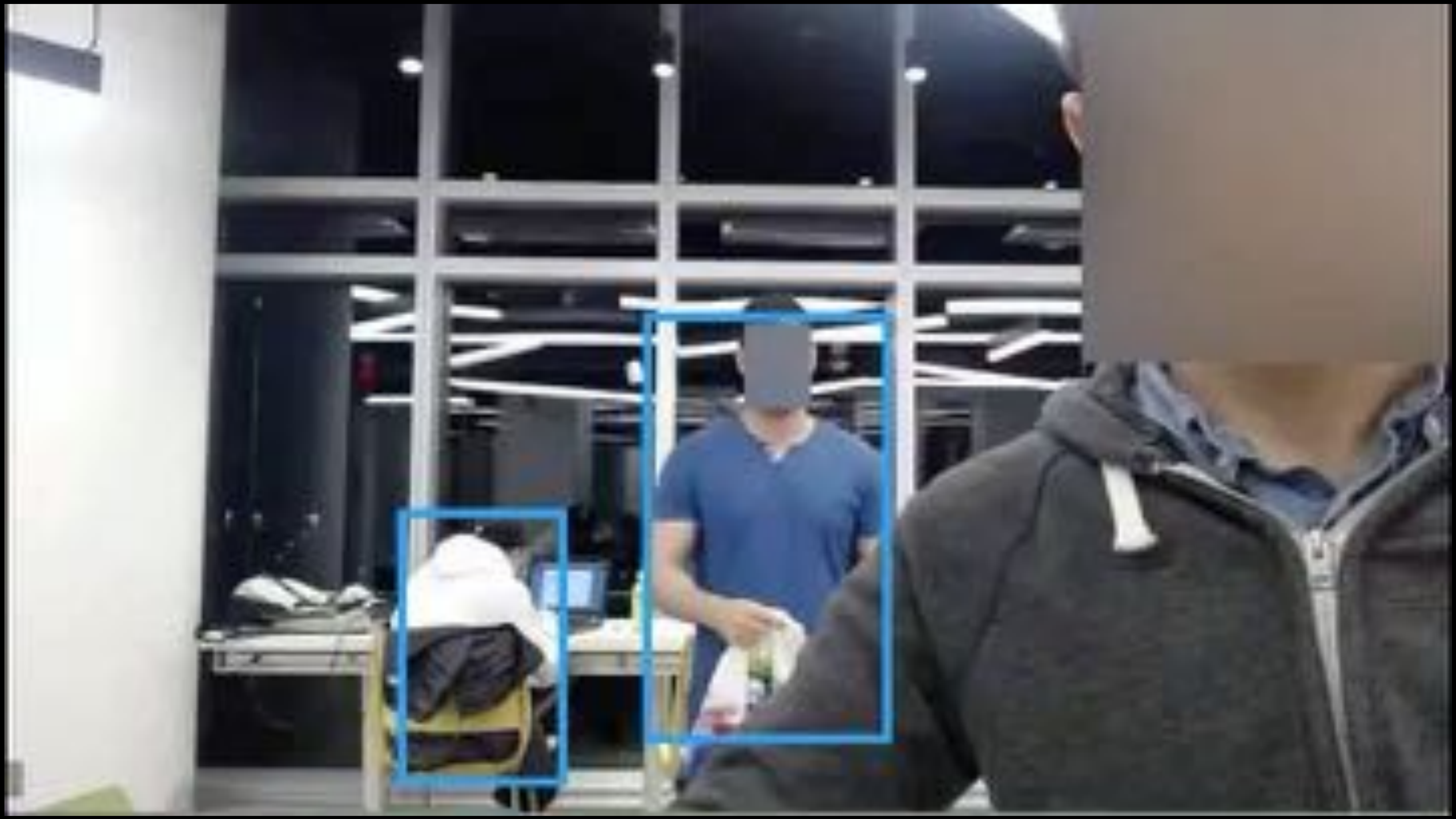
Non-targeted attacks with ensemble

	RMSD	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
-ResNet-152	17.17	0%	0%	0%	0%	0%
-ResNet-101	17.25	0%	1%	0%	0%	0%
-ResNet-50	17.25	0%	0%	2%	0%	0%
-VGG-16	17.80	0%	0%	0%	6%	0%
-GoogLeNet	17.41	0%	0%	0%	0%	5%

- \- Model: the model architecture is not included in the white-box ensemble.
- Ensemble further decreases the accuracy on adversarial examples, and decreases the perturbation magnitude.

Invisibility Cloak

- https://www.cs.umd.edu/~tomg/projects/invisible/
- Adversarial attack on YOLO v2 person detector



References

- 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]
- Finding the Weakest Link in Person Detectors, D. Parikh, and C. L. Zitnick, CVPR, 2011.
- Gehler and Nowozin, On Feature Combination for Multiclass Object Classification, ICCV'09
- 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

- S. Lazebnik, C. Schmid, and J. Ponce, Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories, CVPR 2006
- 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
- 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

- 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
- 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)
- 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

- 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).
- M. Ranzato, Y. Boureau, Y. LeCun. "Sparse Feature Learning for Deep Belief Networks". Advances in Neural Information Processing Systems 20 (NIPS 2007).
- 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.
- 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

- Ruslan Salakhutdinov and Geoffrey Hinton, Deep Boltzmann Machines, 12th International Conference on Artificial Intelligence and Statistics (2009).
- 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.
- 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

- 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.
- R. Girshick, P. Felzenszwalb, D. McAllester, Object Detection with Grammar Models, NIPS 2011
- 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)
- 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

- Long Zhu, Chenxi Lin, Haoda Huang, Yuanhao Chen, Alan Yuille. Unsupervised Structure Learning: Hierarchical Recursive Composition, Suspicious Coincidence and Competitive Exclusion. ECCV 2008.
- 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)
- 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.
- Ruslan Salakhutdinov and Geoffrey Hinton, Deep Boltzmann Machines, 12th International Conference on Artificial Intelligence and Statistics (2009).

References

- 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