Deep Learning

Visualizing and Understanding Convolutional Networks

Christopher Funk

Pennsylvania State University

February 23, 2015

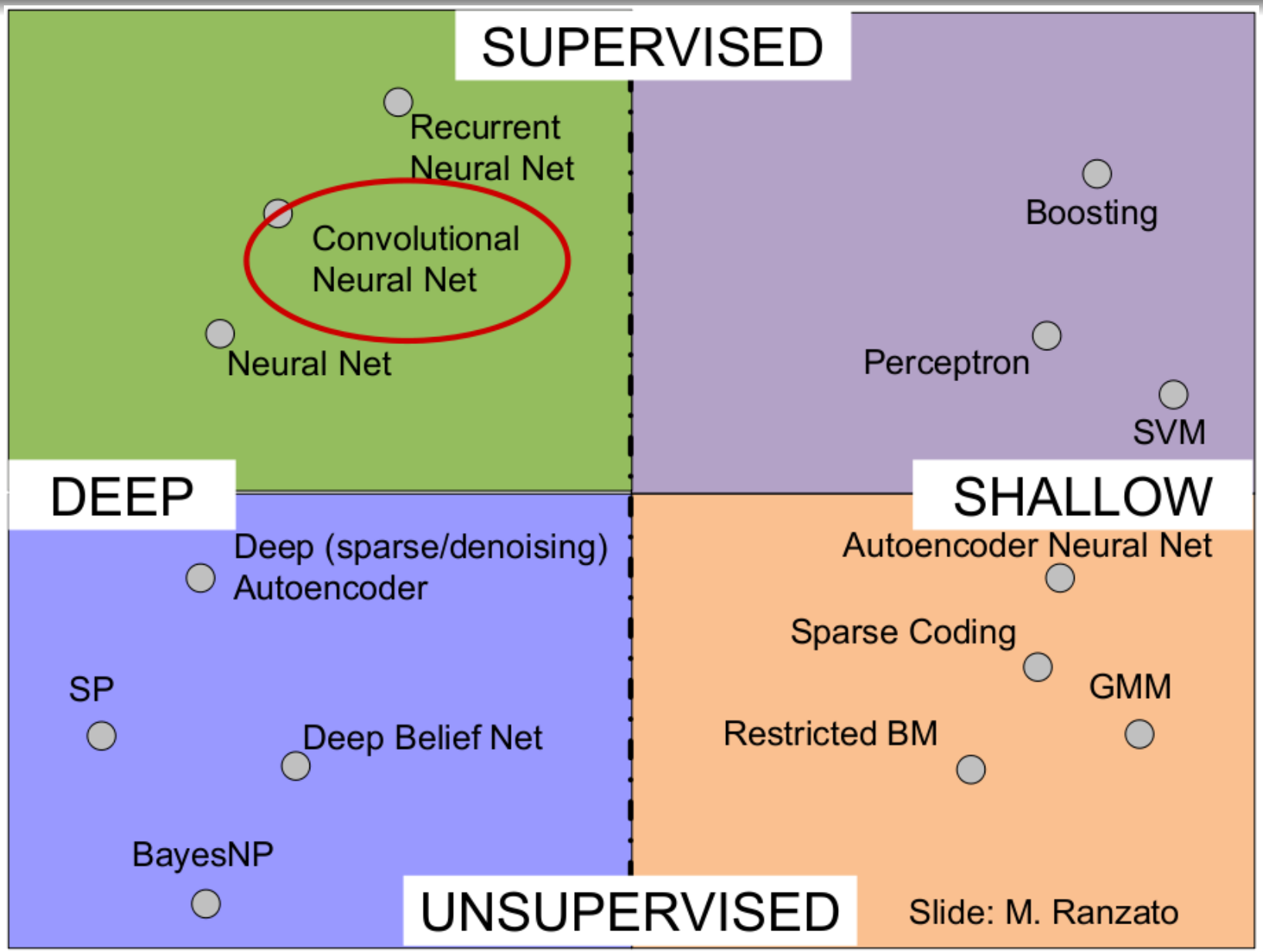
Some Slide Information taken from Pierre Sermanet (Google) presentation on Deep Learning and Computer Vision

Background

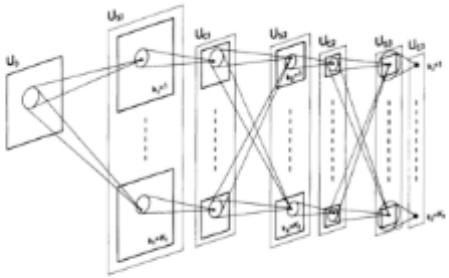
Deep learning is a machine learning technique which utilizes multiple layers of Neural Networks.
The basic Building Blocks are:

- Multiple Layers
- Bias
  Input Weighting (and not the weighting between "Neurons")
- Pooling Operation between layers
  Pooling together inputs which reduces the size of the input between layers. Can be overlapped like in the Alex Network
- ReLUs (Rectified Linear Units)
  Modeling a neuron's output and keeping it positive such as $f(x) = \max(0, x)$. Can also be tied to Local Response Normalization (brightness normalization)
- Types of Learning
  - Stochastic Gradient Decent
  - Restricted Boltzmann Machines
  - Auto-encoders

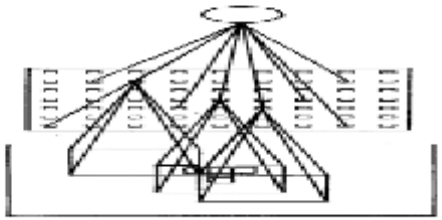
Kinds of Deep Learning Networks



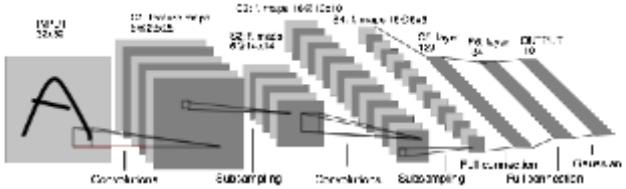
History



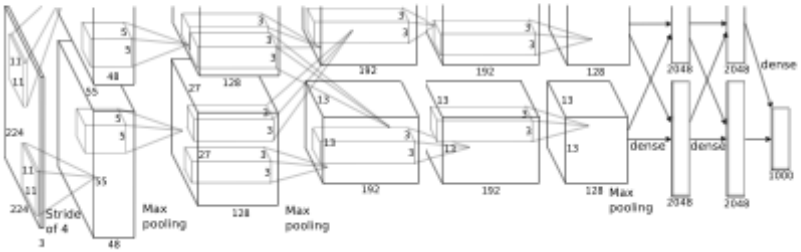
Fukushima 1980
Neocognitron



Rumelhart, Hinton, Williams 1986
"T" versus "C" problem



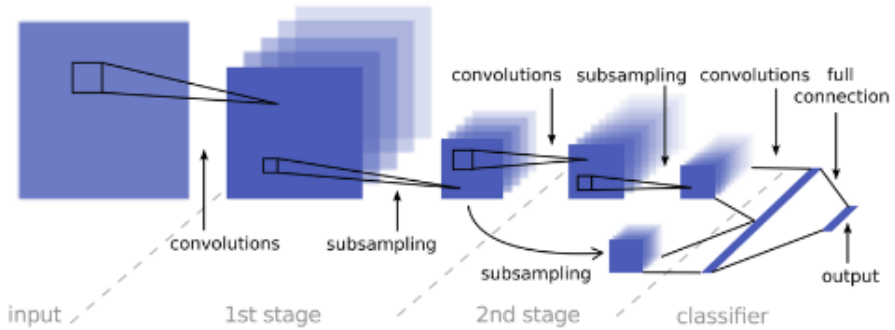
LeCun et al. 1989-1998
Hand-written digit reading

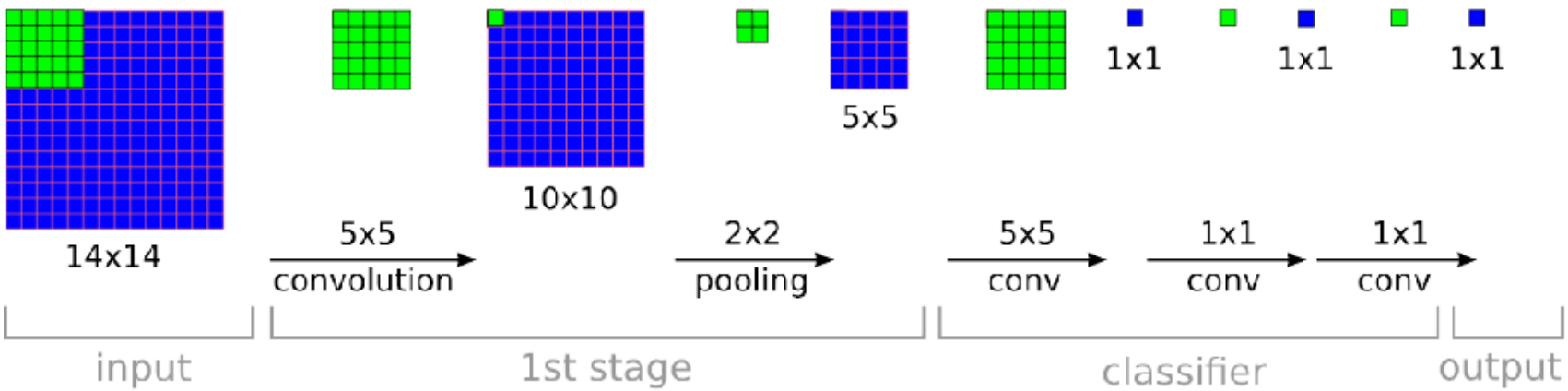


Krizhevksy, Sutskever, Hinton 2012
ImageNet classification breakthrough
"SuperVision" CNN

What is a Convolutional Neuro-Network

- A special type of Neural Net that incorporates **priors about continuous signals**
  - sound / speech (2D signal)
  - images (3D signal)
  - videos (4D signal)
- **Parameters sharing** and **pooling** take advantage of local coherence to learn invariant features
- In its **simplest form**, a ConvNet is just a series of stages of the form:
  - convolution
  - bias
  - non-linearity (ReLu or sigmoid functions)
  - pooling
- Normalization layers (LCN) may be added



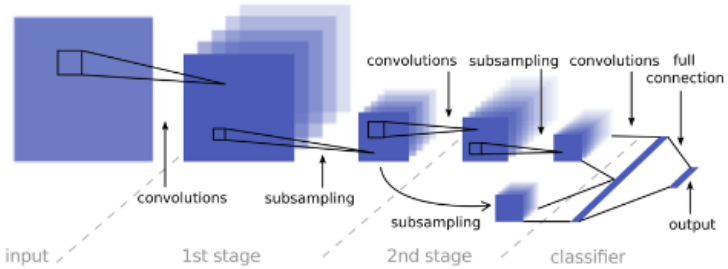


Data Augmentation

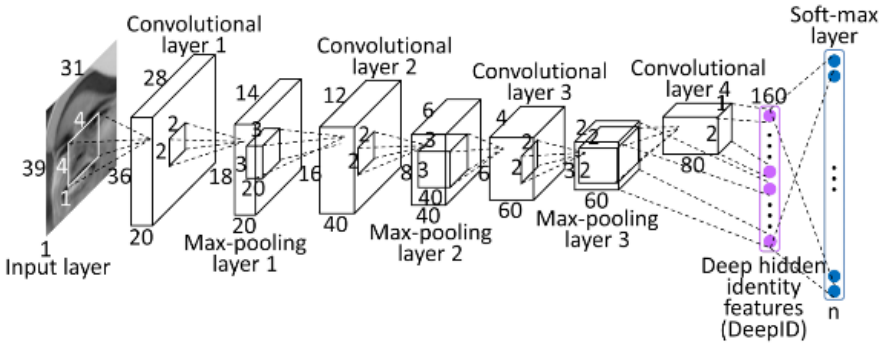
1. Augmenting data is crucial to avoid overfitting and build robustness
2. Ideally you want to cover all deformations occurring in target domain (not more)
   translations
   scales
   rotations
   contrast
   lighting
   colors
   flip
3. If possible, average or max over these transformations at test time
4. Allows for smaller datasets (Imagenet would be far too small without this for the Alex Network)

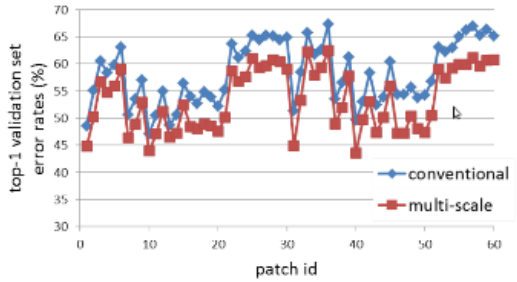
Mutliscale

1. Skip Connections can improve multiscale detection



Task	Single-Stage features	Multi-Stage features	Improvement %
Pedestrians detection (INRIA) [54]	14.26%	9.85%	31%
Traffic Signs classification (GTSRB) [53]	1.80%	0.83%	54%
House Numbers classification (SVHN) [55]	5.54%	5.36%	3.2%





Pedestrian detection with unsupervised multi-stage feature learning. Sermanet, P., Kavukcuoglu, K., Chintala, S., & LeCun, Y. (2013, June). In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*(pp. 3626-3633). IEEE.
Traffic sign recognition with multi-scale convolutional networks. Sermanet, Pierre, and Yann LeCun. *Neural Networks (IJCNN), The 2011 International Joint Conference on*. IEEE, 2011.
Convolutional neural networks applied to house numbers digit classification. Sermanet, Pierre, Soumith Chintala, and Yann LeCun. *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, 2012.
Deep Learning Face Representation from Predicting 10,000 Classes. Sun, Yi, Xiaogang Wang, and Xiaoou Tang.

Spatio-Temporal Features

1. If you know where you need to scale then you can do use multiple streams

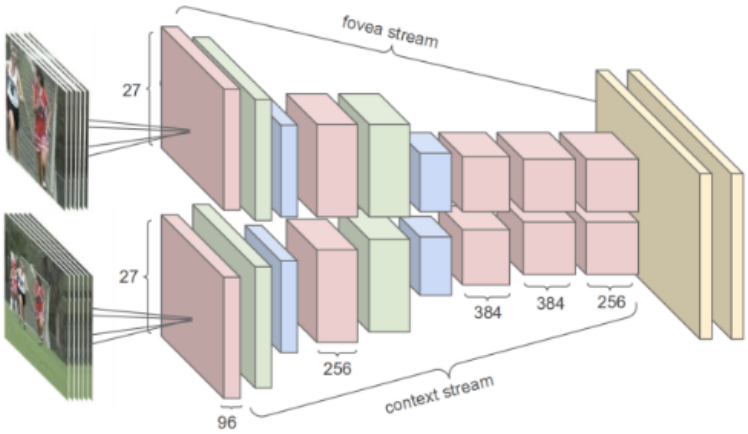


Figure 2: Multiresolution CNN architecture. Input frames are fed into two separate streams of processing: a *context stream* that models low-resolution image and a *fovea stream* that processes high-resolution center crop. Both streams consist of alternating convolution (red), normalization (green) and pooling (blue) layers. Both streams converge to two fully connected layers (yellow).

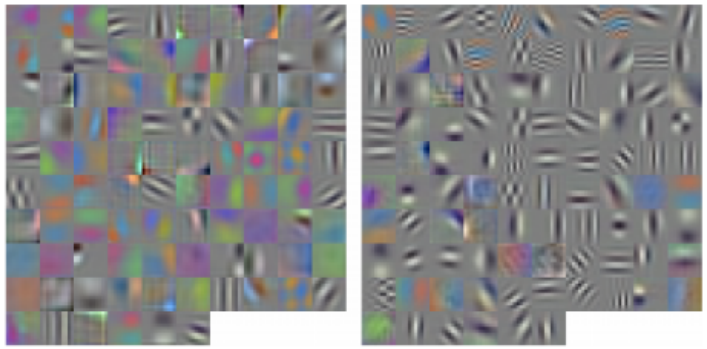
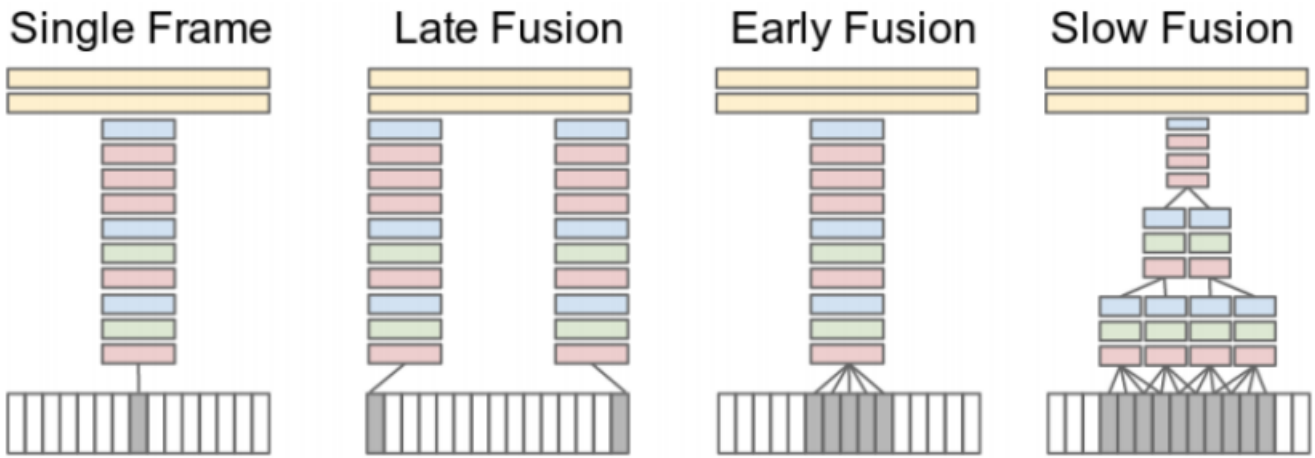


Figure 3: Filters learned on first layer of a multiresolution network. Left: context stream, Right: fovea stream. Notably, the fovea stream learns grayscale, high-frequency features while the context stream models lower frequencies and colors. GIFs of moving video features can be found on our website (linked on first page).

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., & Fei-Fei, L. (2014). **Large-scale video classification with convolutional neural networks.** In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Different kinds of Networks



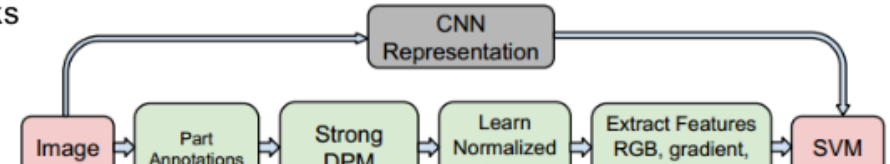
Model	Clip Hit@1	Video Hit@1	Video Hit@5
Feature Histograms + Neural Net	-	55.3	-
Single-Frame	41.1	59.3	77.7
Single-Frame + Multires	**42.4**	**60.0**	**78.5**
Single-Frame Fovea Only	30.0	49.9	72.8
Single-Frame Context Only	38.1	56.0	77.2
Early Fusion	38.9	57.7	76.8
Late Fusion	40.7	59.3	78.7
Slow Fusion	**41.9**	**60.9**	**80.2**
CNN Average (Single+Early+Late+Slow)	41.4	63.9	82.4

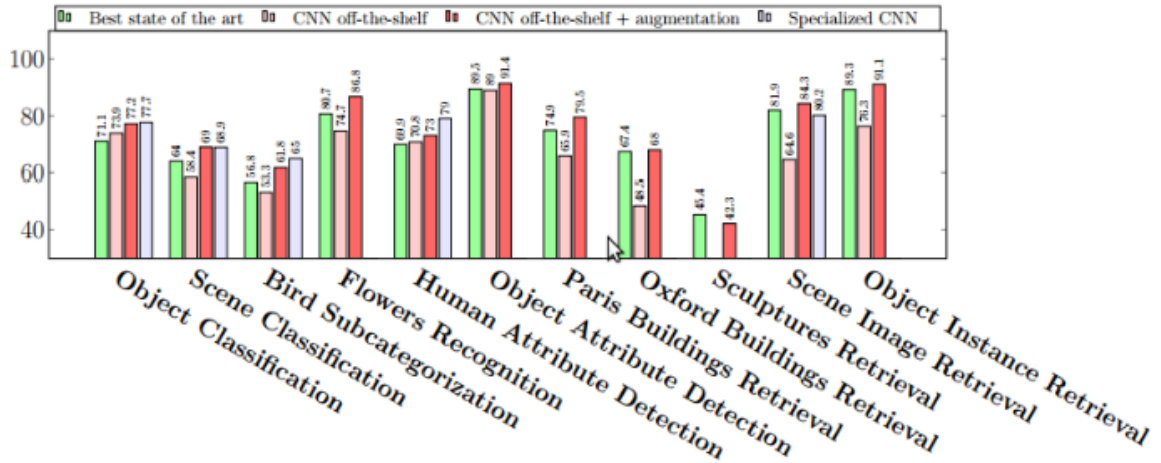
Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., & Fei-Fei, L. (2014). **Large-scale video classification with convolutional neural networks.** In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

ImageNet pre-training

1. Leveraging Previously Trained datasets (Transfer Learning)

- **Labeled data is rare for detection:** leverage large classification labeled datasets for pre-training.
- **ImageNet Classification pretraining + fine-tuning on a different task** has been shown to work very well by many people.
  - in particular [Razavian'14] took the off-the-shelf convnet **OverFeat + SVM classifier** on top and obtained many state-of-the-art or competitive results on 10+ datasets and visual tasks





CNN Features off-the-shelf: an Astounding Baseline for Recognition. Razavian, Ali Sharif, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. *arXiv preprint arXiv:1403.6382* (2014).

ImageNet pre-training

1. Size of your dataset dictates the number of levels (or at least the approach)

- **Capacity must match the problem at hand**
  - 60M-parameters model has a capacity designed for ImageNet-scale data
  - one cannot train such model on a small dataset: e.g. 6k bird dataset in [Branson'14]
  - ImageNet pre-training will ensure **general features as a starting point**

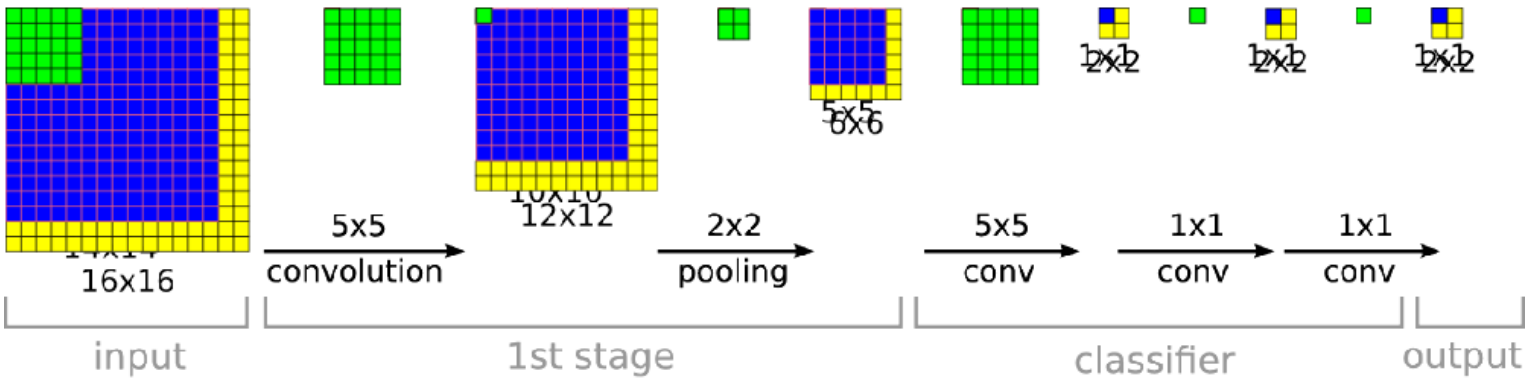
- **Fine-tuning**
  - requires **lowering the learning rate** to avoid forgetting pre-training or use **different learning rates for the pre-trained and new layers**
  - [Branson'14] propose a **2-step fine-tuning method** that improves accuracy
    i. only train the weights of the new layer(s)
    ii. train all weights

Bird Species Categorization Using Pose Normalized Deep Convolutional Nets. Branson, Steve, Grant Van Horn, Serge Belongie, and Pietro Perona. *arXiv preprint arXiv:1406.2952* (2014).

Why are CNNs so good?

1. Size of your dataset dictates the number of levels (or at least the approach)

- **Sharing parameters** is good
  - taking advantage of local coherence to learn a more efficient representation:
    - no redundancy
    - translation invariance
    - slight rotation invariance with pooling
- **Efficient for detection:**
  - all computations are shared
  - can handle varying input sizes (no need to relearn weights for new sizes)
- **ConvNets are convolutional all the way up** including fully connected layers



Pedestrian Detection

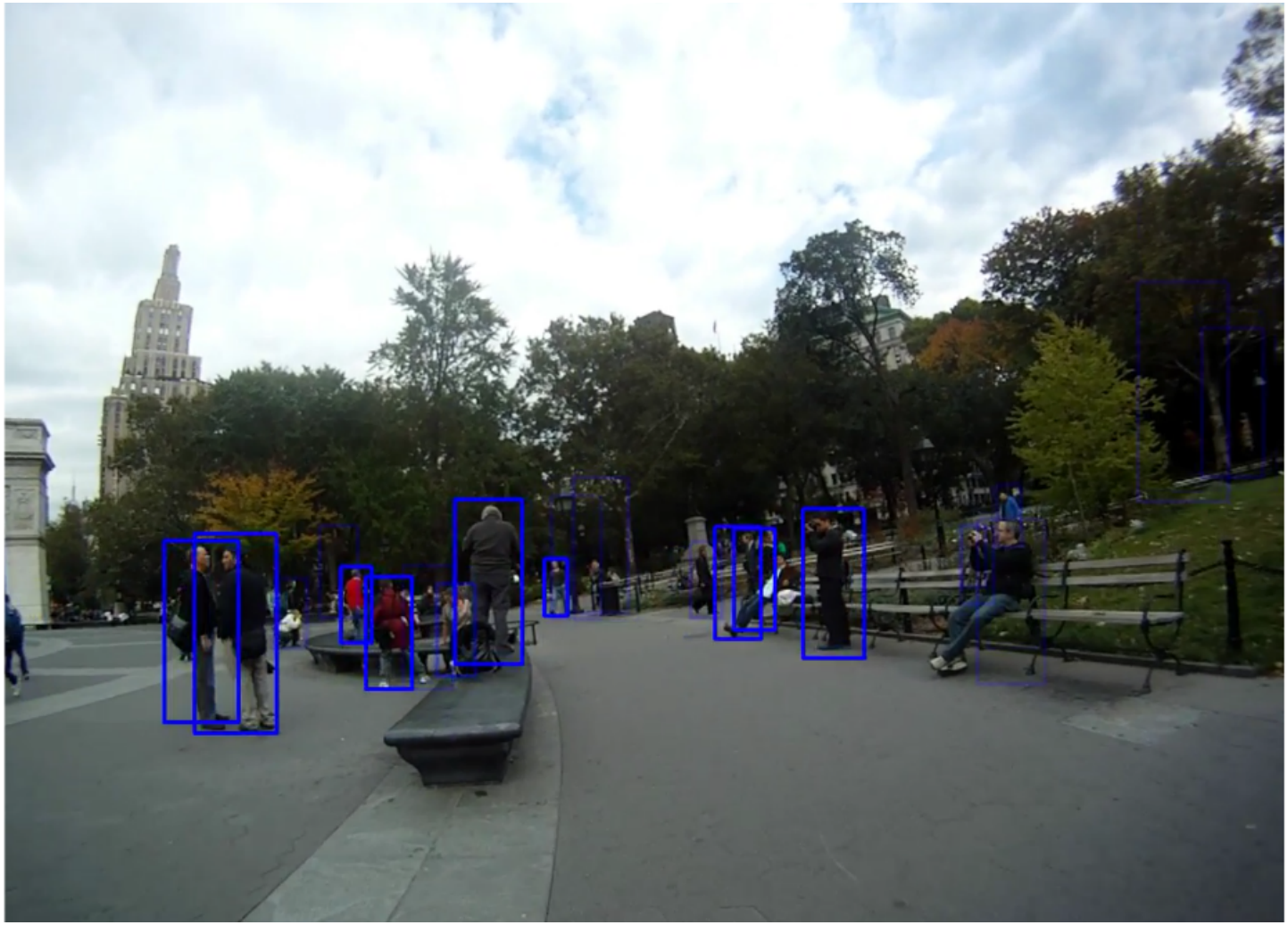


Table of Contents

Visualizing and Understanding Convolutional Networks

Authors: Matthew D. Zeiler and Rob Fergus.
Dept. of Computer Science, Courant Institute, NYU.

Abstract: Large Convolutional Network models have recently demonstrated impressive classification performance on the ImageNet benchmark (Krizhevsky et al., 2012). However there is no clear understanding of why they perform so well, or how they might be improved. In this paper we address both issues. We introduce a novel visualization technique that gives insight into the function of intermediate feature layers and the operation of the classifier. Used in a diagnostic role, these visualizations allow us to find model architectures that outperform Krizhevsky et al. on the ImageNet classification benchmark. We also perform an ablation study to discover the performance contribution from different model layers. We show our ImageNet model generalizes well to other datasets: when the softmax classifier is retrained, it convincingly beats the current state-of-the-art results on Caltech-101 and Caltech-256 datasets.

Purpose

- The purpose of the paper is to propose a way of visualizing the inter workings of the Convolutional Neuro-Networks called a Deconvolutional Network (deconvnet)
  - Understand Deep Learning in general
  - Understand how the specific network is working
- Reverse the flow of the network to visualize the learned elements. Not a generative projection from the model!
- They are showing the highest activation levels of the filter from a validation set

Basic Opperations

- Unpooling
  The unpooling step is to reverse of the pooling operation between layers. This normally is a one way function so they have to include variable (called switches) which represent which elements the pooling operation is selecting to move to the next layer.
- Rectification
  This operation makes sure that the output of the reverse direction is non-negative. This is done in the normal feed forward network stage as well.
- Filtering
  The CNN uses convolution with learned filters at beginning stage of each layer. To invert this, they use a transpose of the same filter but apply rectified maps to this element (instead of a different part of the layer)

Visualization of Deconvnet

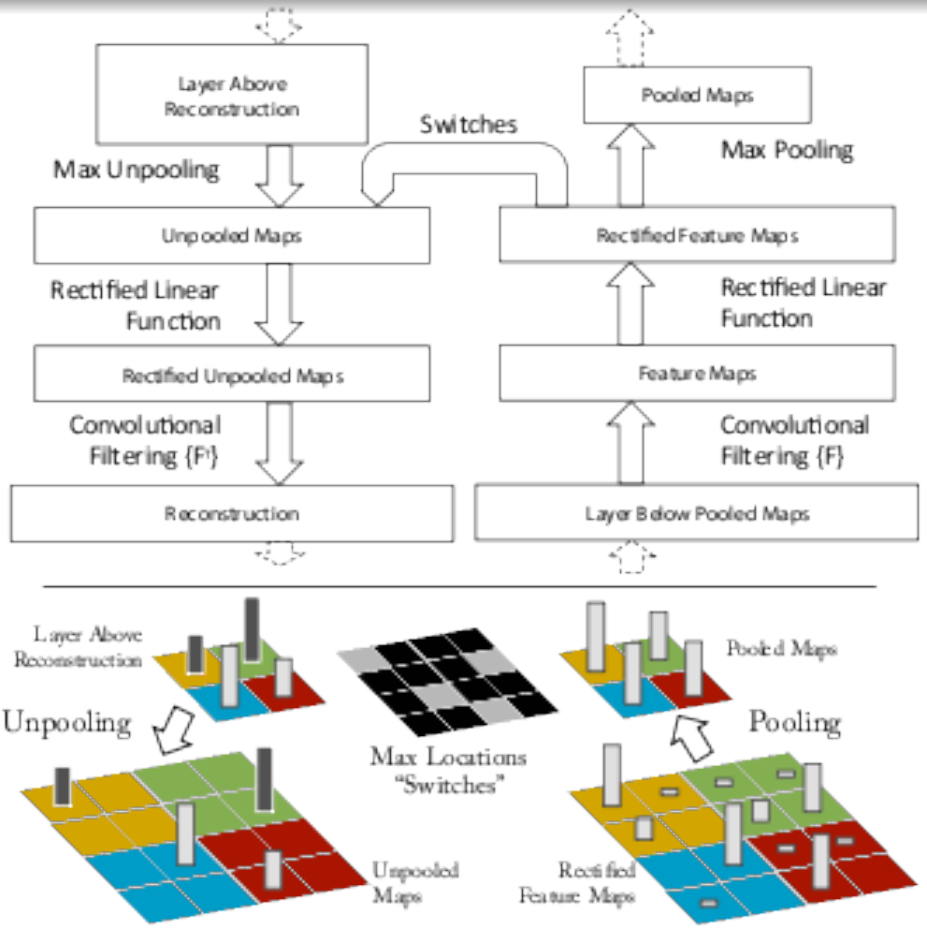


Figure 1. Top: A deconvnet layer (left) attached to a convnet layer (right). The deconvnet will reconstruct an approximate version of the convnet features from the layer beneath. Bottom: An illustration of the unpooling operation in the deconvnet, using *switches* which record the location of the local max in each pooling region (colored zones) during pooling in the convnet.

The Network

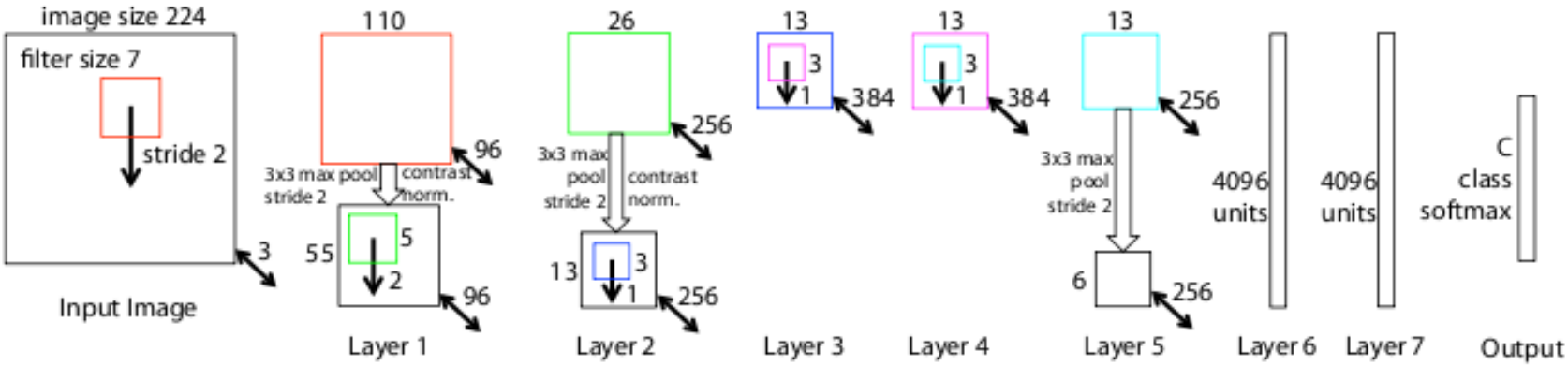


Figure 3. Architecture of our 8 layer convnet model. A 224 by 224 crop of an image (with 3 color planes) is presented as the input. This is convolved with 96 different 1st layer filters (red), each of size 7 by 7, using a stride of 2 in both x and y. The resulting feature maps are then: (i) passed through a rectified linear function (not shown), (ii) pooled (max within 3x3 regions, using stride 2) and (iii) contrast normalized across feature maps to give 96 different 55 by 55 element feature maps. Similar operations are repeated in layers 2,3,4,5. The last two layers are fully connected, taking features from the top convolutional layer as input in vector form ($6 \cdot 6 \cdot 256 = 9216$ dimensions). The final layer is a C-way softmax function, C being the number of classes. All filters and feature maps are square in shape.

Training

They use a dense amount of connections for the 3, 4, and 5 layers unlike the Alex Network which used sparse connections because of less hardware restrictions
They trained on ImageNet 2012 training set (1.3 Million Images / 1000 different classes)
Image Processing Steps

- Resize smallest dimension to 256
- Cropping the Center 256x256 region
- Subtracting the per-pixel Mean (across all images)
- 10 different sub-crops of size 244x244 (corners + center with(out) horizontal flips)

Training Parameters Continued

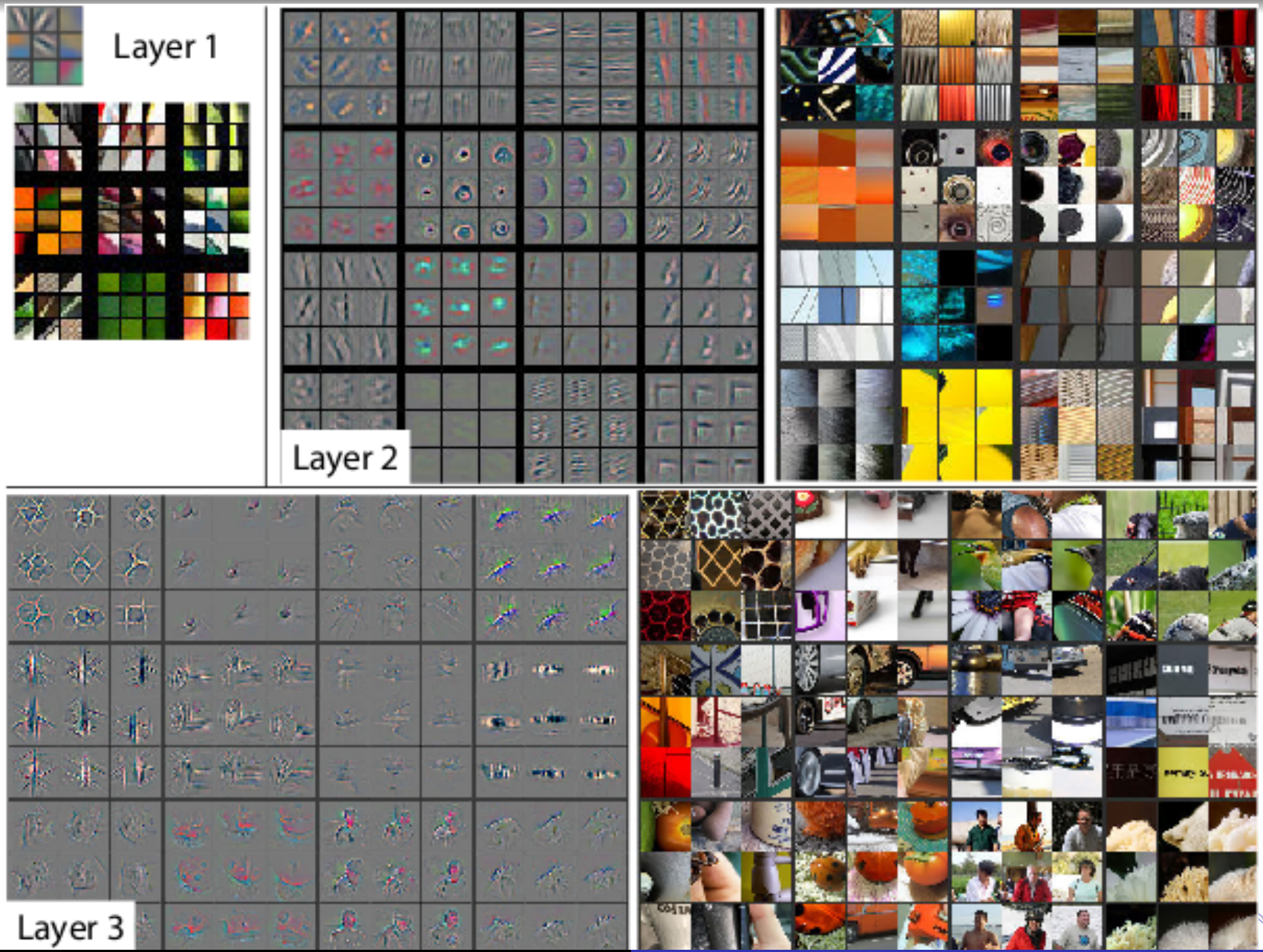
Learning Parameters

- Stochastic gradient descent with a mini-batch size of 128 to update parameters
- Starting with a learning rate of 10^{-2} and a momentum term of 0.9
- They anneal the learning rate throughout training manually when the validation error plateaus
- Dropout is used in the fully connected layers (6 and 7) with a rate of 0.5
- All weights are initialized to 10^{-2}

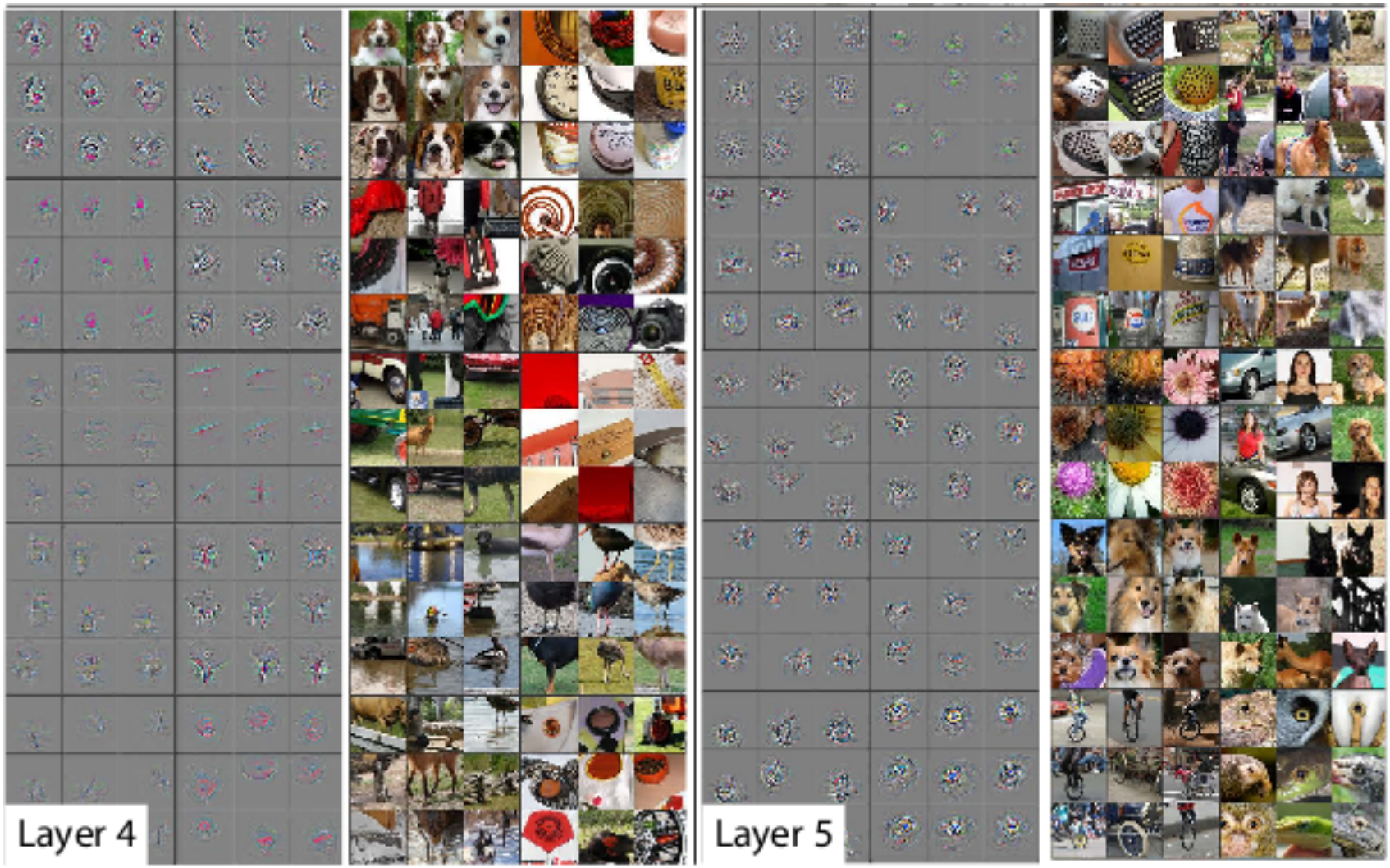
The visualization of the first layers showed that some filters were becoming too dominate so they renormalized layers with a high RMS

It took them 12 days of training on a single GPU using 70 epochs

Final Visualizations Part 1



Final Visualizations Part 2



Final Visualizations Part 3

Figure 2. Visualization of features in a fully trained model. For layers 2-5 we show the top 9 activations in a random subset of feature maps across the validation data, projected down to pixel space using our deconvolutional network approach. Our reconstructions are *not* samples from the model: they are reconstructed patterns from the validation set that cause high activations in a given feature map. For each feature map we also show the corresponding image patches. Note: (i) the the strong grouping within each feature map, (ii) greater invariance at higher layers and (iii) exaggeration of discriminative parts of the image, e.g. eyes and noses of dogs (layer 4, row 1, cols 1). Best viewed in electronic form.

- l5 r1 c2 patches have little in common but the background
- l3 r1 c1 complex invariances capturing similar textures
- l4 significant variation but class specific
- l5 entire obj with significant pose variation

Figure Evolution During Training

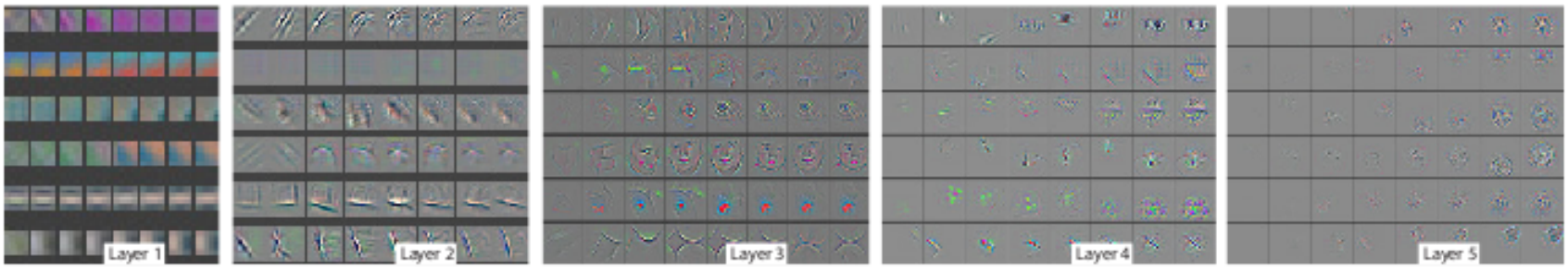


Figure 4. Evolution of a randomly chosen subset of model features through training. Each layer's features are displayed in a different block. Within each block, we show a randomly chosen subset of features at epochs [1,2,5,10,20,30,40,64]. The visualization shows the strongest activation (across all training examples) for a given feature map, projected down to pixel space using our deconvnet approach. Color contrast is artificially enhanced and the figure is best viewed in electronic form.

Figure Invariance

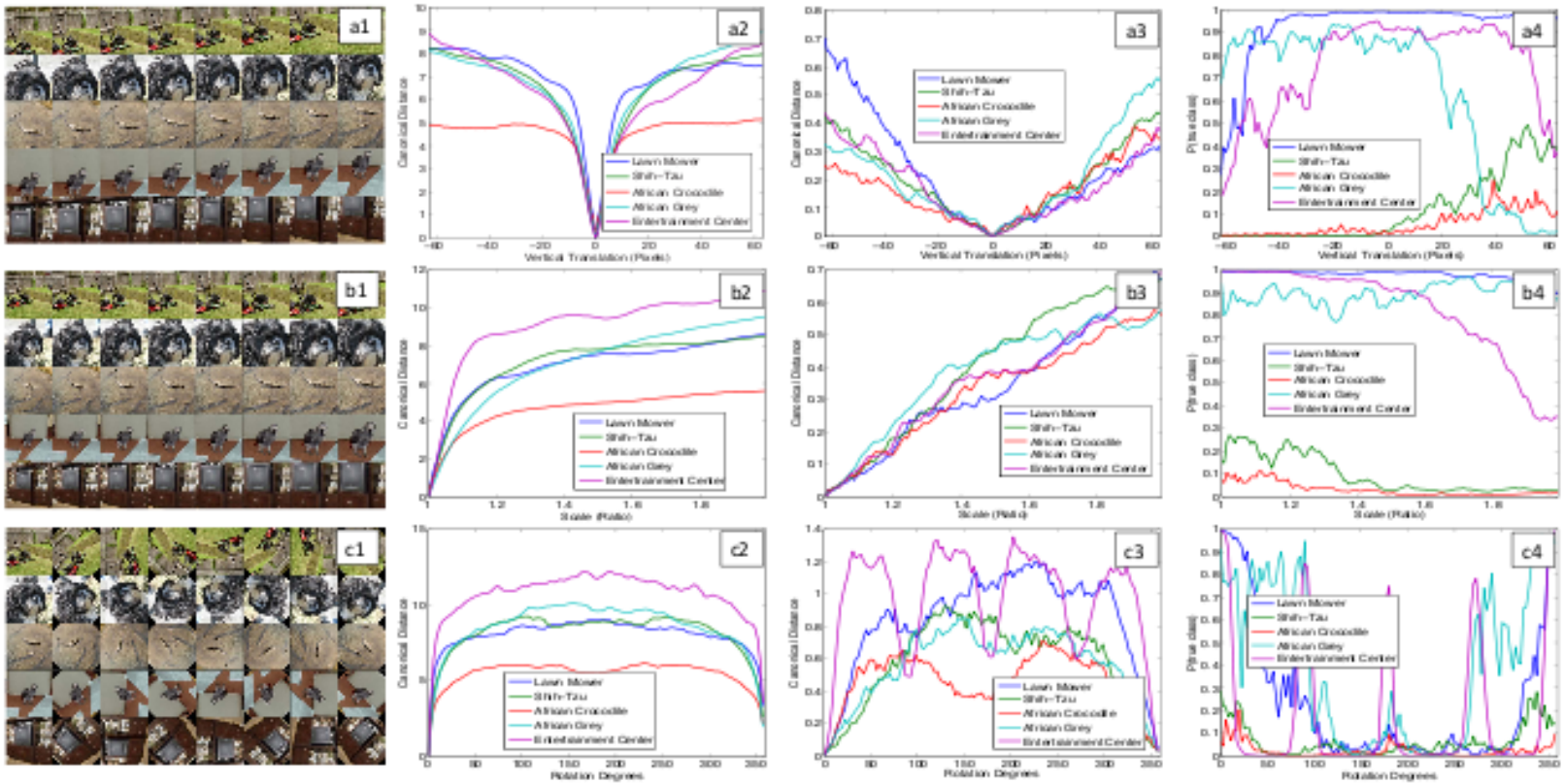


Figure 5. Analysis of vertical translation, scale, and rotation invariance within the model (rows a-c respectively). Col 1: 5 example images undergoing the transformations. Col 2 & 3: Euclidean distance between feature vectors from the original and transformed images in layers 1 and 7 respectively. Col 4: the probability of the true label for each image, as the image is transformed.

Architecture Selection

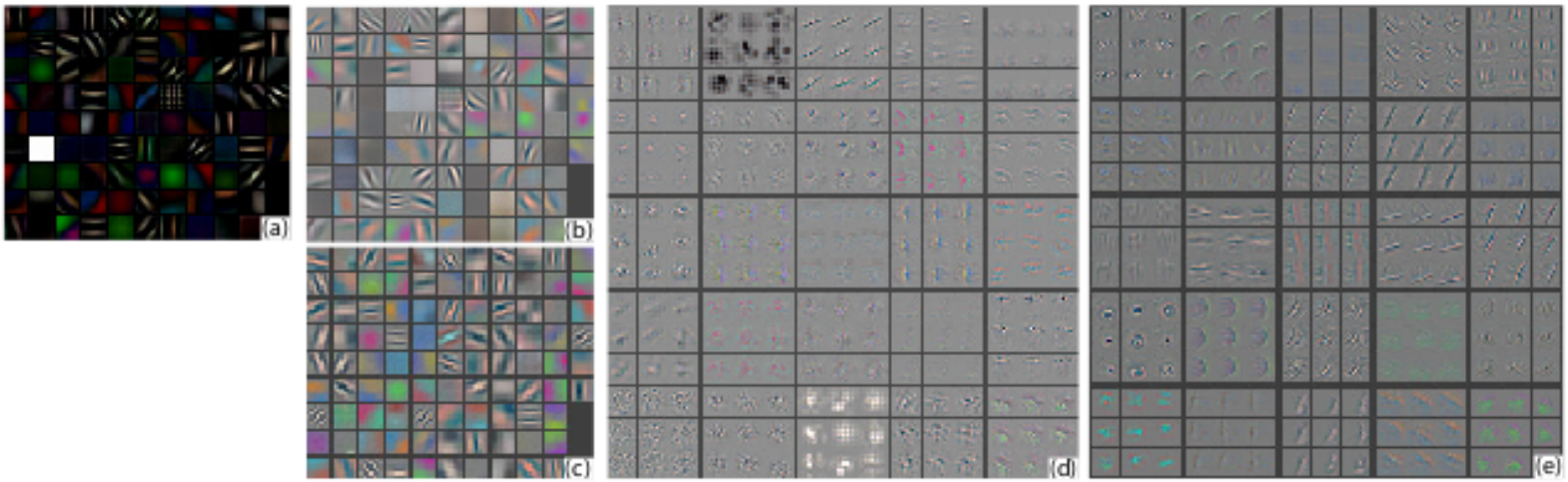


Figure 6. (a): 1st layer features without feature scale clipping. Note that one feature dominates. (b): 1st layer features from (Krizhevsky et al., 2012). (c): Our 1st layer features. The smaller stride (2 vs 4) and filter size (7x7 vs 11x11) results in more distinctive features and fewer "dead" features. (d): Visualizations of 2nd layer features from (Krizhevsky et al., 2012). (e): Visualizations of our 2nd layer features. These are cleaner, with no aliasing artifacts that are visible in (d).

Occlusion Sensitivity

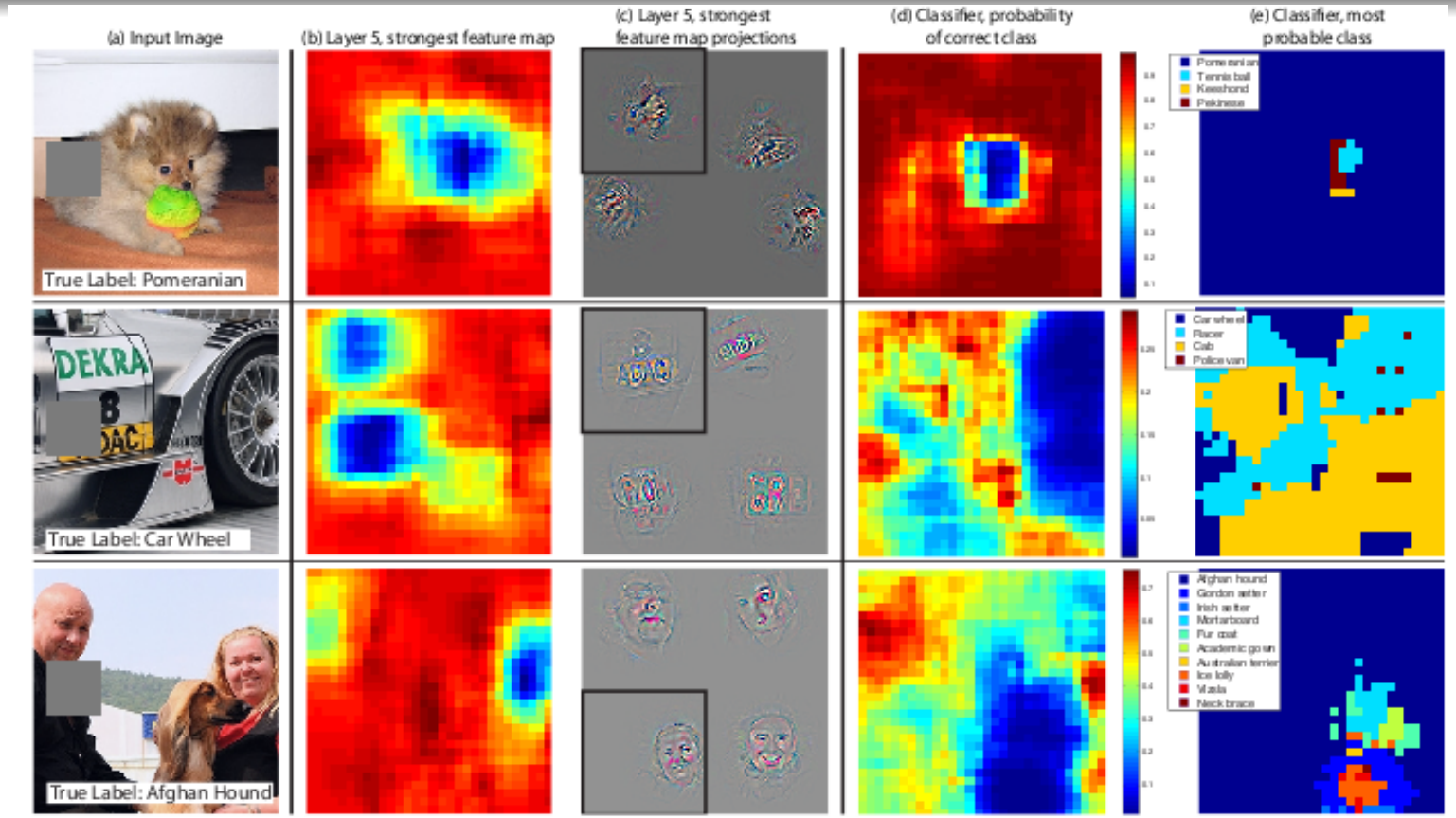


Figure 7. Three test examples where we systematically cover up different portions of the scene with a gray square (1st column) and see how the top (layer 5) feature maps ((b) & (c)) and classifier output ((d) & (e)) changes. (b): for each position of the gray scale, we record the total activation in one layer 5 feature map (the one with the strongest response in the unoccluded image). (c): a visualization of this feature map projected down into the input image (black square), along with visualizations of this map from other images. The first row example shows the strongest feature to be the dog's face. When this is covered-up the activity in the feature map decreases (blue area in (b)). (d): a map of correct class probability, as a function of the position of the gray square. E.g. when the dog's face is obscured, the probability for "pomeranian" drops significantly. (e): the most probable label as a function of occluder position. E.g. in the 1st row, for most locations it is "pomeranian", but if the dog's face is obscured but not the ball, then it predicts "tennis ball". In the 2nd example, text on the car is the strongest feature in layer 5, but the classifier is most sensitive to the wheel. The 3rd example contains multiple objects. The strongest feature in layer 5 picks out the faces, but the classifier is sensitive to the dog (blue region in (d)), since it uses multiple feature maps.

Correspondence Analysis

Figure 8. Images used for correspondence experiments. Col 1: Original image. Col 2,3,4: Occlusion of the right eye, left eye, and nose respectively. Other columns show examples of random occlusions.

Occlusion Location	Mean Feature Sign Change Layer 5	Mean Feature Sign Change Layer 7
Right Eye	0.067 ± 0.007	0.069 ± 0.015
Left Eye	0.069 ± 0.007	0.068 ± 0.013
Nose	0.079 ± 0.017	0.069 ± 0.011
Random	0.107 ± 0.017	0.073 ± 0.014

Table 1. Measure of correspondence for different object parts in 5 different dog images. The lower scores for the eyes and nose (compared to random object parts) show the model implicitly establishing some form of correspondence of parts at layer 5 in the model. At layer 7, the scores are more similar, perhaps due to upper layers trying to discriminate between the different breeds of dog.

Results

- First they Replicated the Alex Network results, than surpassed the best with error rate of 14.8%
- Then they played around with tweaking the network (results bottom right)
- Convolutional Part is important for best results

Error %	Val Top-1	Val Top-5	Test Top-5
(Gunji et al., 2012)	-	-	26.2
(Krizhevsky et al., 2012), 1 convnet	40.7	18.2	– –
(Krizhevsky et al., 2012), 5 convnets	38.1	16.4	16.4
(Krizhevsky et al., 2012)*, 1 convnets	39.0	16.6	– –
(Krizhevsky et al., 2012)*, 7 convnets	36.7	15.4	15.3
Our replication of (Krizhevsky et al., 2012), 1 convnet	40.5	18.1	– –
1 convnet as per Fig. 3	38.4	16.5	– –
5 convnets as per Fig. 3 – (a)	36.7	15.3	15.3
1 convnet as per Fig. 3 but with layers 3,4,5: 512,1024,512 maps – (b)	37.5	16.0	16.1
6 convnets, (a) & (b) combined	**36.0**	**14.7**	**14.8**

Table 2. ImageNet 2012 classification error rates. The * indicates models that were trained on both ImageNet 2011 and 2012 training sets.

Error %	Train Top-1	Val Top-1	Val Top-5
Our replication of (Krizhevsky et al., 2012), 1 convnet	35.1	40.5	18.1
Removed layers 3,4	41.8	45.4	22.1
Removed layer 7	27.4	40.0	18.4
Removed layers 6,7	27.4	44.8	22.4
Removed layer 3,4,6,7	71.1	71.3	50.1
Adjust layers 6,7: 2048 units	40.3	41.7	18.8
Adjust layers 6,7: 8192 units	26.8	40.0	18.1
Our Model (as per Fig. 3)	33.1	38.4	16.5
Adjust layers 6,7: 2048 units	38.2	40.2	17.6
Adjust layers 6,7: 8192 units	22.0	38.8	17.0
Adjust layers 3,4,5: 512,1024,512 maps	18.8	**37.5**	**16.0**
Adjust layers 6,7: 8192 units and Layers 3,4,5: 512,1024,512 maps	**10.0**	38.3	16.9

Table 3. ImageNet 2012 classification error rates with various architectural changes to the model of (Krizhevsky et al., 2012) and our model (see Fig. 3).

Results Generalizaed on Caltech-101/256

- Training on 15 or 30 randomly selected images per class and test on 50 per class
- The results showed how pre-training on a large dataset increase accuracy tremendously
- Just need 6 Caltech-256 training images to beat the next best which needs 10 times as many

# Train	Acc % 15/class	Acc % 30/class
(Bo et al., 2013)	–	81.4 ± 0.33
(Jianchao et al., 2009)	73.2	84.3
Non-pretrained convnet	22.8 ± 1.5	46.5 ± 1.7
ImageNet-pretrained convnet	$\mathbf{83.8 \pm 0.5}$	$\mathbf{86.5 \pm 0.5}$

Table 4. Caltech-101 classification accuracy for our convnet models, against two leading alternate approaches.

# Train	Acc % 15/class	Acc % 30/class	Acc % 45/class	Acc % 60/class
(Sohn et al., 2011)	35.1	42.1	45.7	47.9
(Bo et al., 2013)	40.5 ± 0.4	48.0 ± 0.2	51.9 ± 0.2	55.2 ± 0.3
Non-pretr.	9.0 ± 1.4	22.5 ± 0.7	31.2 ± 0.5	38.8 ± 1.4
ImageNet-pretr.	$\mathbf{65.7 \pm 0.2}$	$\mathbf{70.6 \pm 0.2}$	$\mathbf{72.7 \pm 0.4}$	$\mathbf{74.2 \pm 0.3}$

Table 5. Caltech 256 classification accuracies.

Results Generalizaed on Pascal 2012

- They could not beat the state of the art here overall (but they did for some classes)
- Might be because of the difference in the datasets
- 3.2% from best reported result but who's counting

Acc %	[A]	[B]	Ours	Acc %	[A]	[B]	Ours
Airplane	92.0	**97.3**	96.0	Dining tab	63.2	**77.8**	67.7
Bicycle	74.2	**84.2**	77.1	Dog	68.9	83.0	**87.8**
Bird	73.0	80.8	**88.4**	Horse	78.2	**87.5**	86.0
Boat	77.5	85.3	**85.5**	Motorbike	81.0	**90.1**	85.1
Bottle	54.3	**60.8**	55.8	Person	91.6	**95.0**	90.9
Bus	85.2	**89.9**	85.8	Potted pl	55.9	**57.8**	52.2
Car	81.9	**86.8**	78.6	Sheep	69.4	79.2	**83.6**
Cat	76.4	89.3	**91.2**	Sofa	65.4	**73.4**	61.1
Chair	65.2	**75.4**	65.0	Train	86.7	**94.5**	91.8
Cow	63.2	**77.8**	74.4	Tv	77.4	**80.7**	76.1
Mean	74.3	**82.2**	79.0	# won	0	**15**	5

Table 6. PASCAL 2012 classification results, comparing our Imagenet-pretrained convnet against the leading two methods ([A]= (Sande et al., 2012) and [B] = (Yan et al., 2012)).

Feature Analysis

- They add an SVM or softmax classifier at the end of each layer of the Imagenet-pretrained models
- They show how well each additional layer does at classifying the image

	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)	$\mathbf{85.4 \pm 0.4}$	$\mathbf{72.6 \pm 0.1}$

Table 7. Analysis of the discriminative information contained in each layer of feature maps within our ImageNet-pretrained convnet. We train either a linear SVM or softmax on features from different layers (as indicated in brackets) from the convnet. Higher layers generally produce more discriminative features.

Discussion

- Features are far from randomize
- Increasing Invariance while ascending through the levels
- Can use this to debug problems
- Occlusion can be use to localization of objects in the image
- The increase in classification is cause by holistic elements of the model and no specific aspect