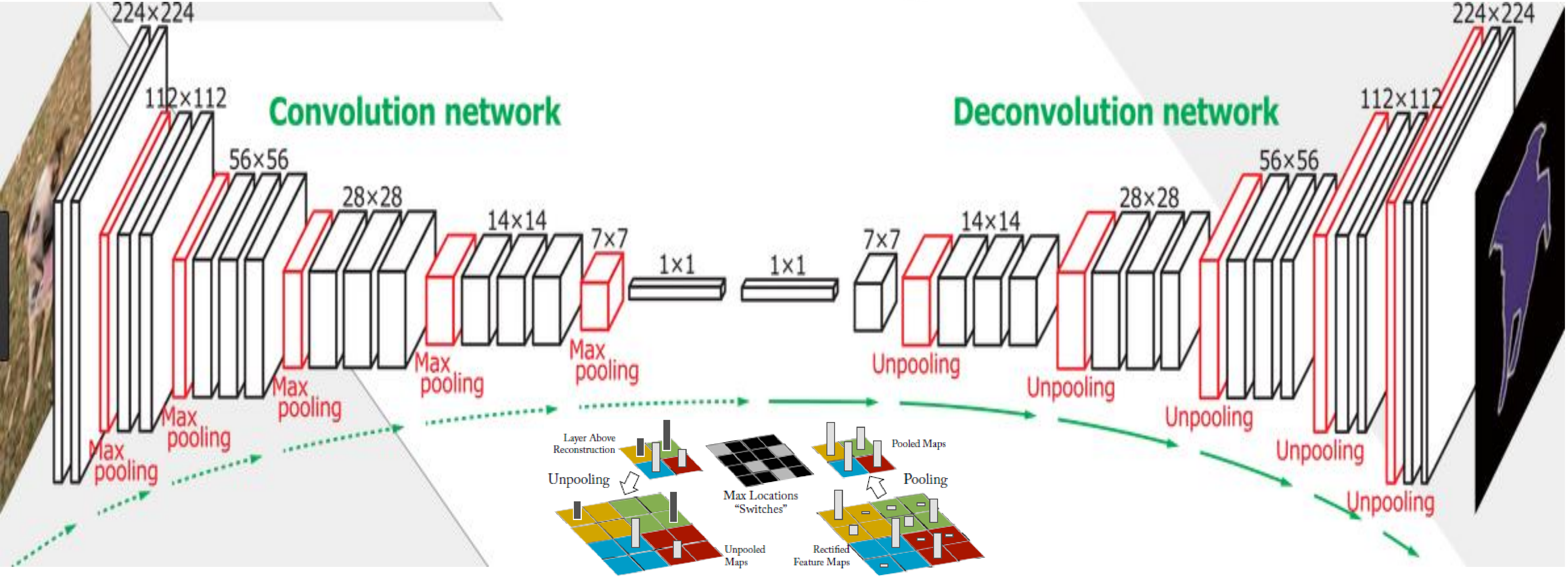
Deep Learning



Article: Visualizing and Understanding Convolutional Networks

Renan Aquino 18/12/2017

The Author’s

Matthew Zeiler, Founder and CEO of Clarifai, is a machine learning Ph.D. and thought leader pioneering the field of applied artificial intelligence (AI). Matt’s groundbreaking research in computer vision alongside renowned machine learning experts Geoff Hinton and Yann LeCun has propelled the image recognition industry from theory to real-world application. Since starting Clarifai in 2013, Matt has evolved his award-winning research into developer-friendly products that allow enterprises to quickly and seamlessly integrate AI into their workflows and customer experiences. Today, Clarifai is the leading independent AI company and “widely seen as one of the most promising [startups] in the crowded, buzzy field of machine learning.”

Matthew Zeiler

Rob Fergus is an Associate Professor of Computer Science at the Courant Institute of Mathematical Sciences, New York University. He is also a Research Scientist at Facebook, working in their AI Research Group. He received a Masters in Electrical Engineering with Prof. Pietro Perona at Caltech, before completing a PhD with Prof. Andrew Zisserman at the University of Oxford in 2005. Before coming to NYU, he spent two years as a post-doc in the Computer Science and Artificial Intelligence Lab (CSAIL) at MIT, working with Prof. William Freeman. He has received several awards including a CVPR best paper prize, a Sloan Fellowship & NSF Career award and the IEEE Longuet-Higgins prize.

Rob Fergus

Introduction

Since their introduction by (LeCun et al., 1989) in the early 1990's, Convolutional Networks (convnets) have demonstrated excellent performance at tasks such as hand-written digit classification and face detection.

Most notably, (Krizhevsky et al., 2012) show record beating performance on the ImageNet 2012 classification benchmark, with their convnet model achieving an error rate of 16.4%, compared to the 2nd place result of 26.1%.

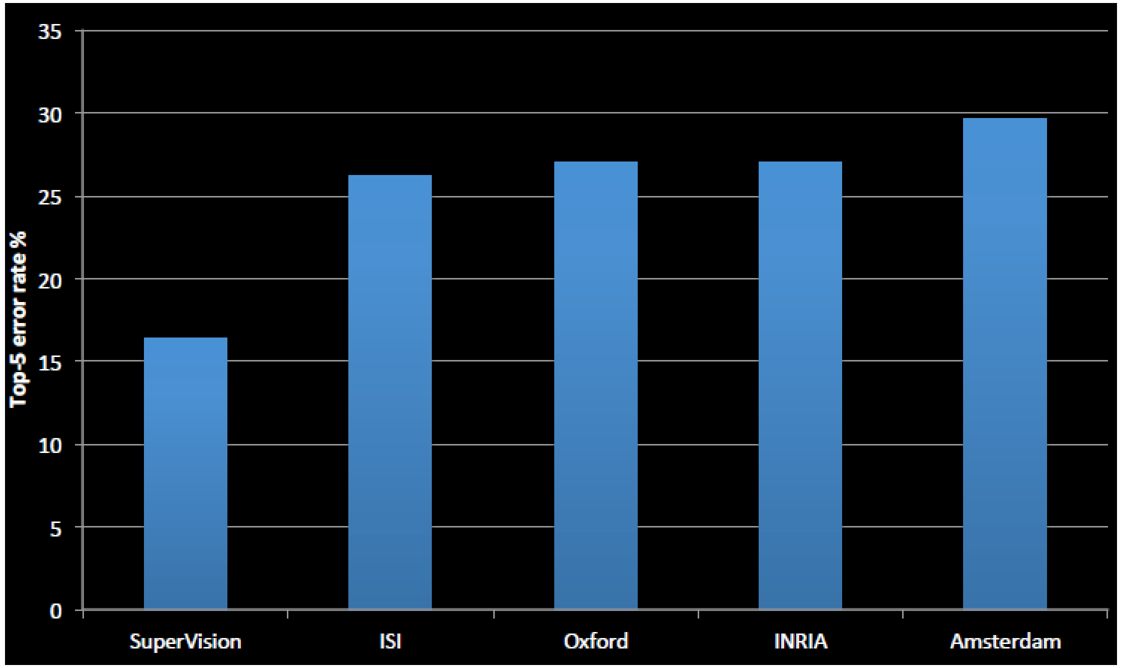
Several factors are responsible for this renewed interest in convnet models:

- ✓ The availability of much larger training sets, with millions of labeled examples;
- ✓ Powerful GPU implementations;
- ✓ Better model regularization strategies, such as Dropout (Hinton et al., 2012).

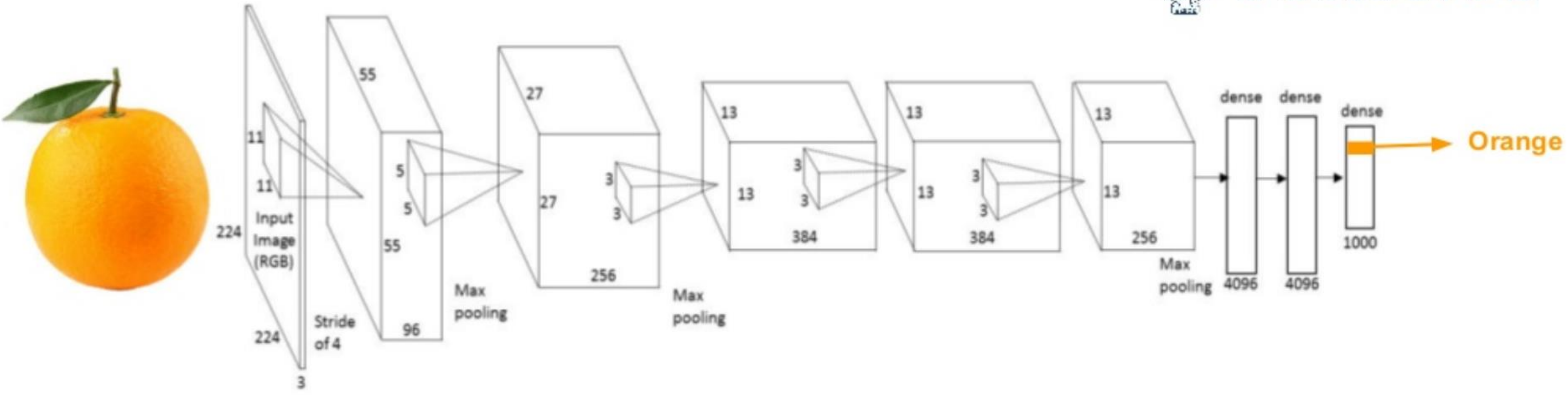
The visualization technique we propose uses a multi-layered Deconvolutional Network (deconvnet), as proposed by (Zeiler et al., 2011), to project the feature activations back to the input pixel space.

A sensitivity analysis of the classifier output by occluding portions of the input image, revealing which parts of the scene are important for classification.

ImageNet Classification 2012

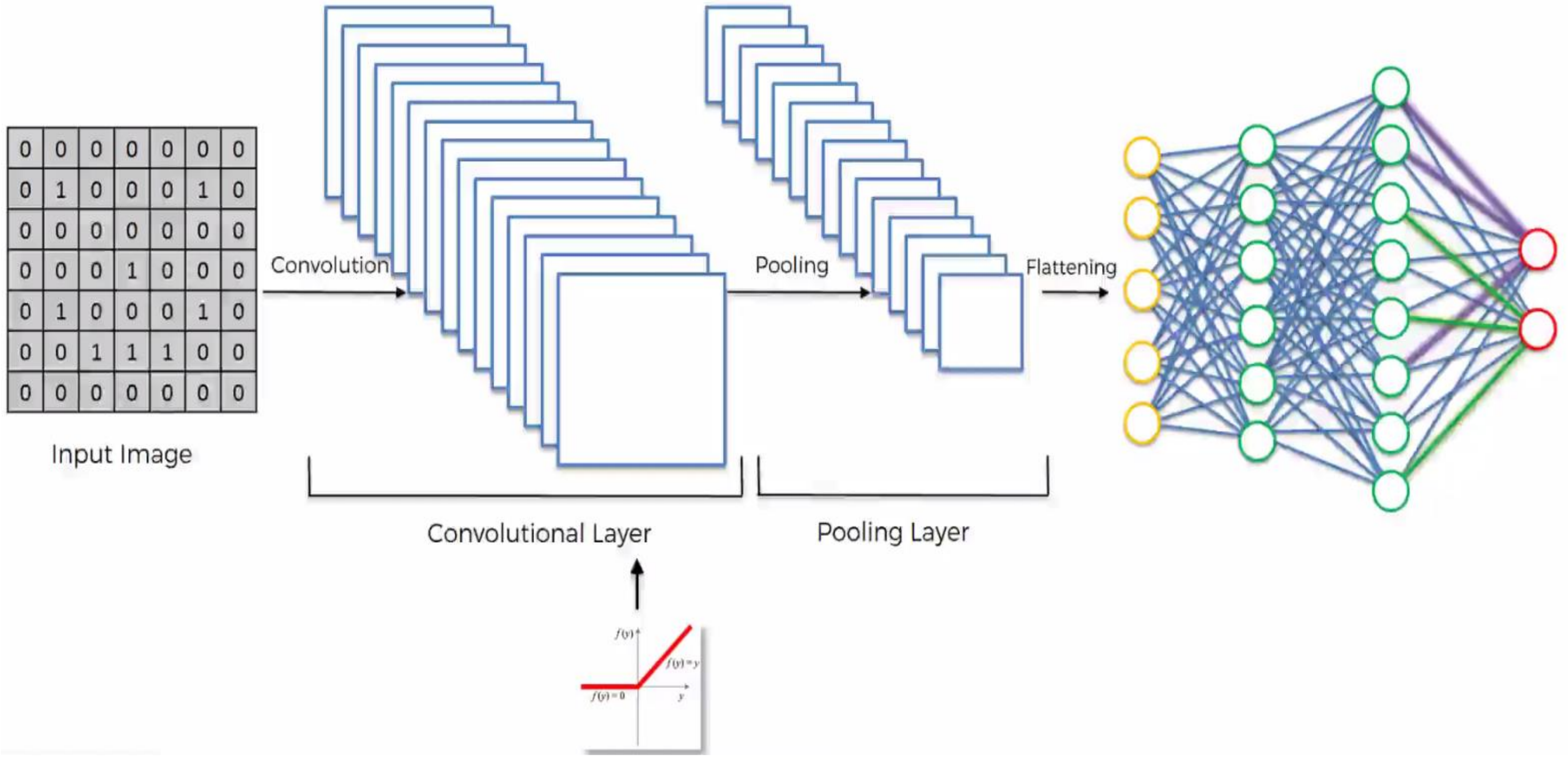


AlexNet (Supervision)

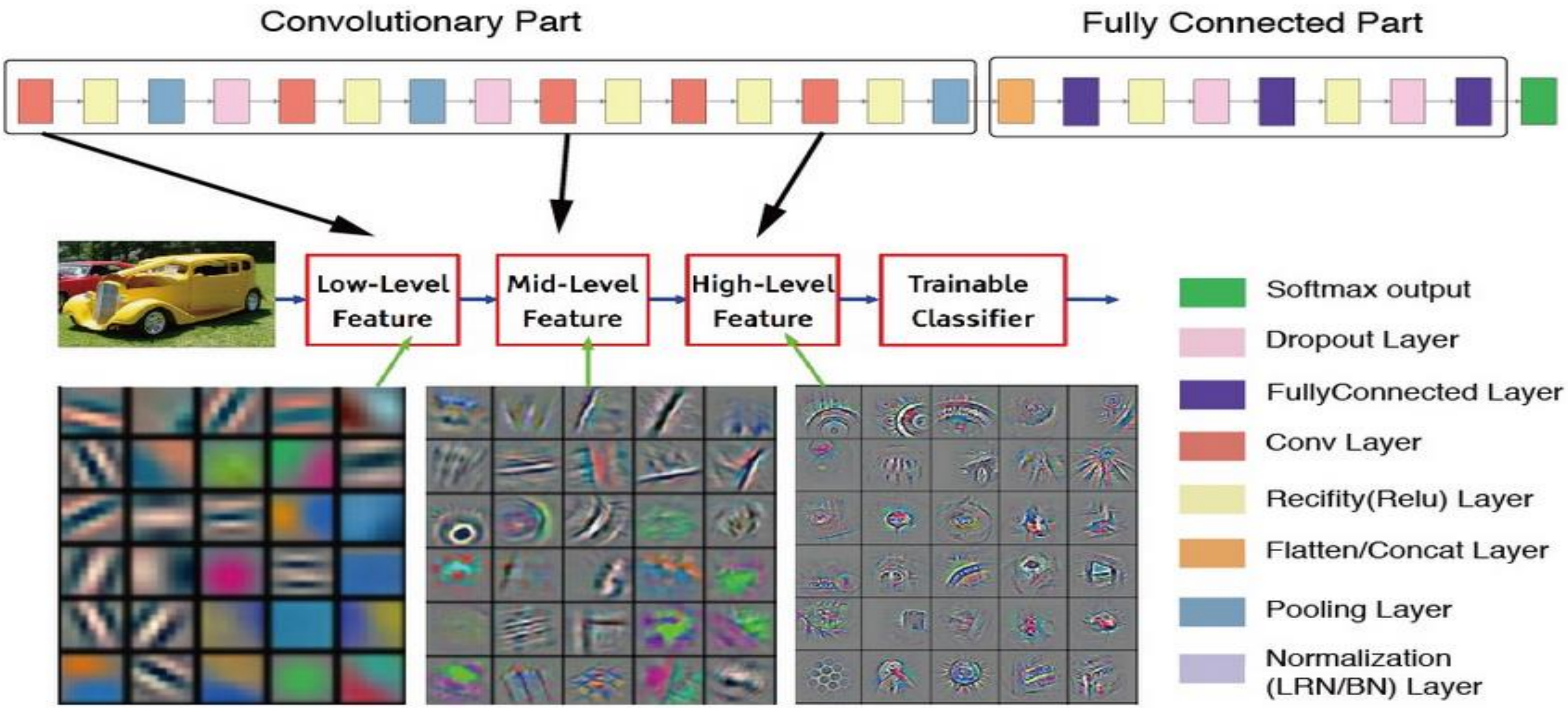


A Krizhevsky, I Sutskever, GE Hinton "Imagenet classification with deep convolutional neural networks" Part of: Advances in Neural Information Processing Systems 25 (NIPS 2012)

Introduction

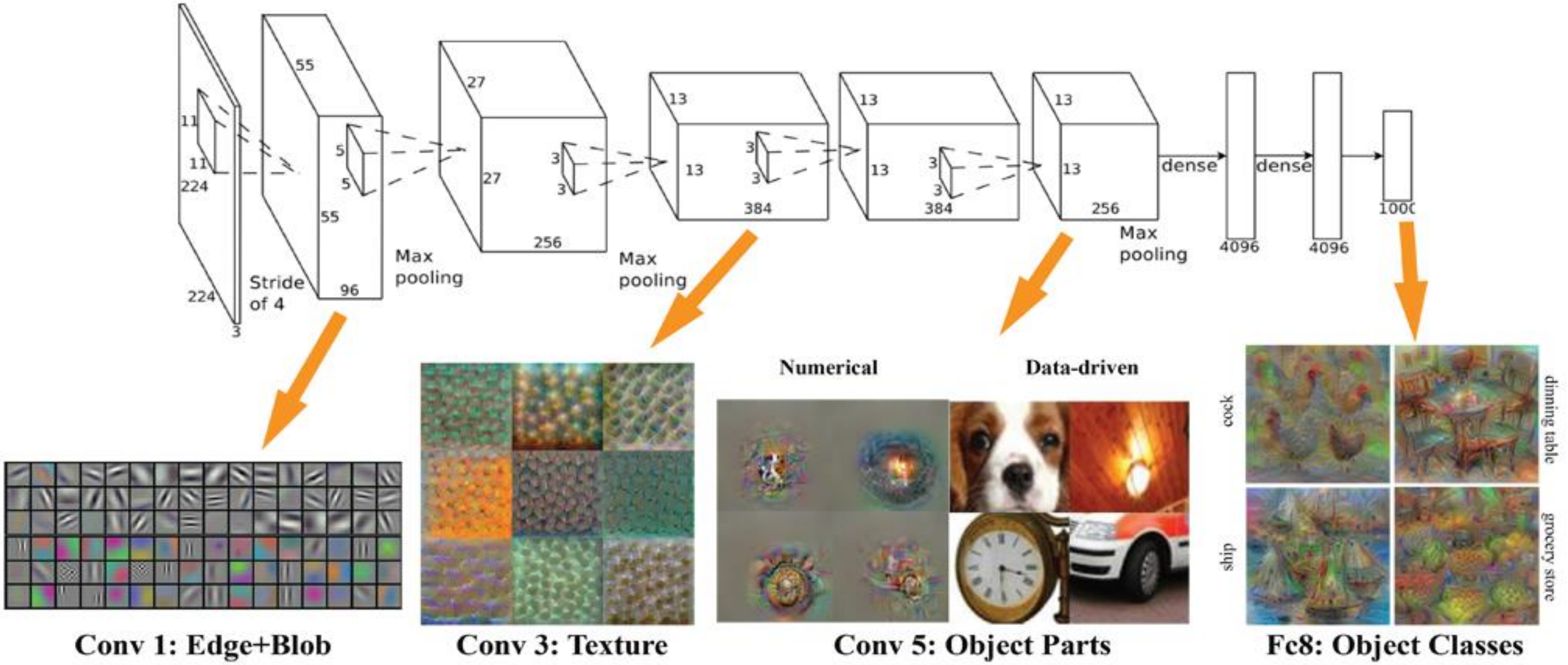


Introduction



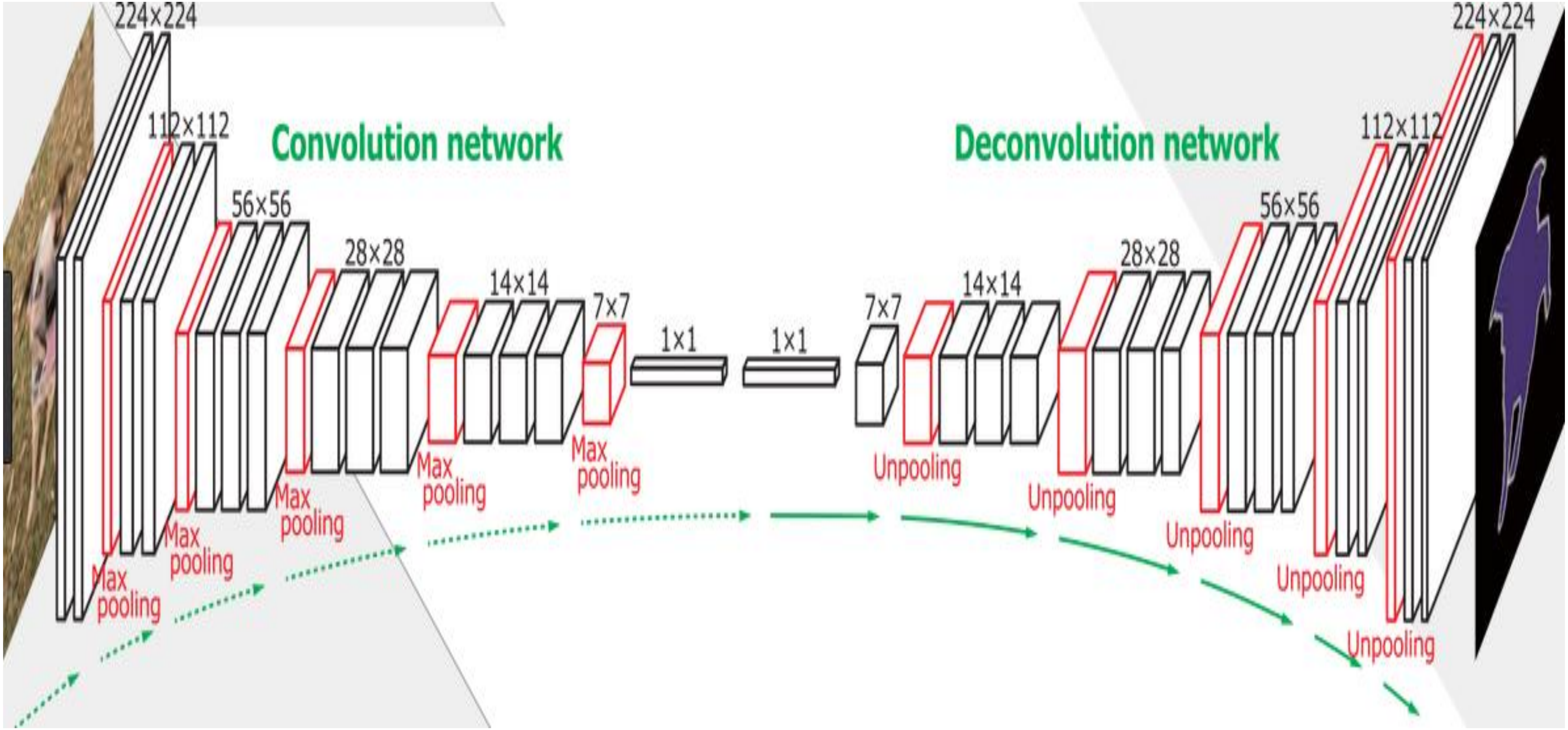
Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Introduction



AlexNet / VGG-F network visualized by **mNeuron**.

Approach



224x224
112x112
56x56
28x28
14x14
7x7
1x1
1x1
7x7
14x14
28x28
56x56
112x112
224x224
Convolution network
Deconvolution network
Max pooling
Max pooling
Max pooling
Max pooling
Max pooling
Unpooling
Unpooling
Unpooling
Unpooling
Unpooling

Approach

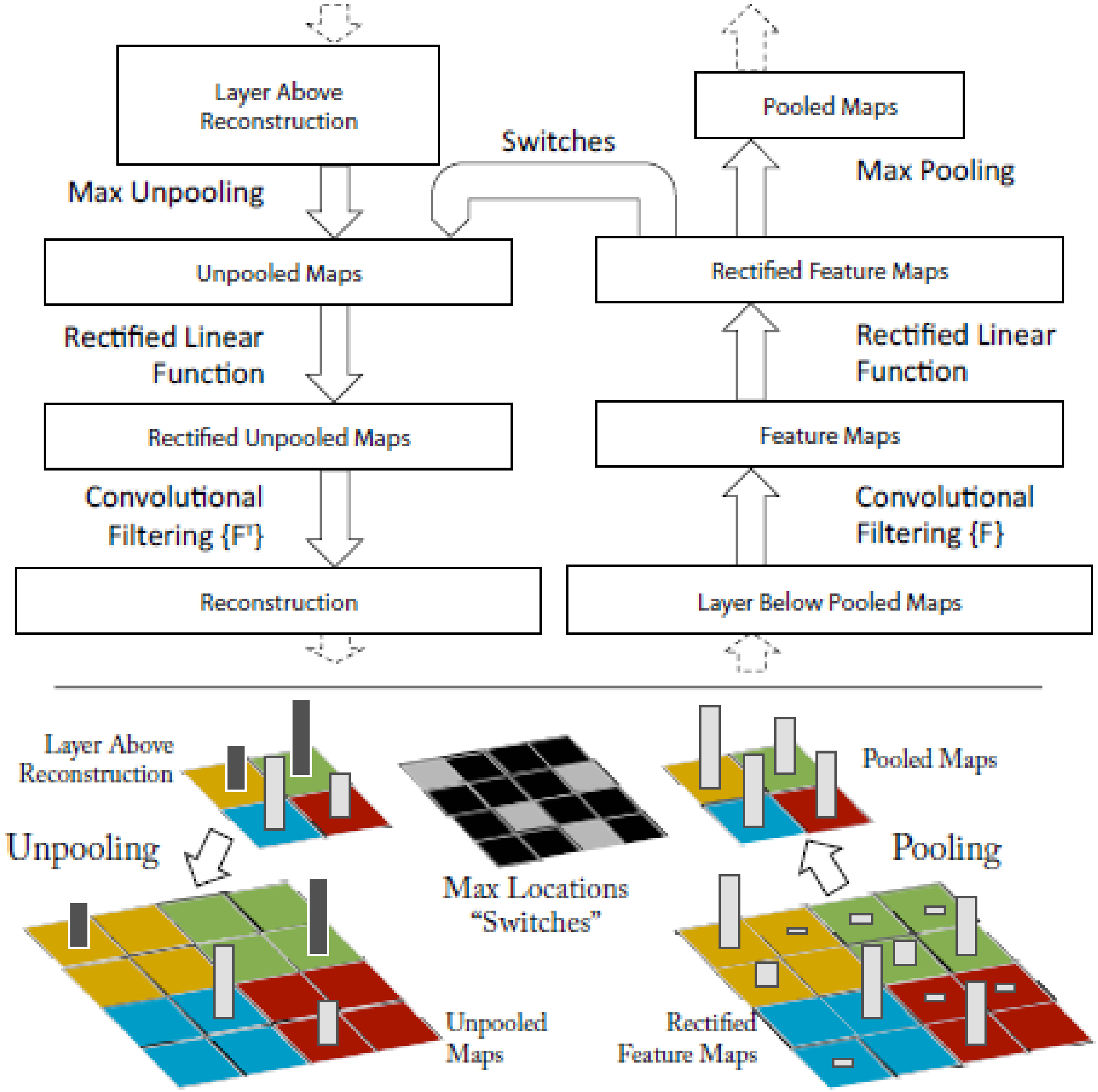
Each layer consists:

- ✓ Convolution of the previous layer output (or, in the case of the 1st layer, the input image) with a set of learned filters;
- ✓ Passing the responses through a rectified linear function (relu(x) = max(x; 0));
- ✓ Max pooling over local neighborhoods;
- ✓ A local contrast operation that normalizes the responses across feature maps.

The top few layers of the network are conventional fully-connected networks and the final layer is a softmax classifier.

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.



Training Details

The model was trained on the ImageNet 2012 training set (1.3 million images, spread over 1000 different classes).

Each RGB image was preprocessed by resizing the smallest dimension to 256, cropping the center 256x256 region, subtracting the per-pixel mean (across all images) and then using 10 different sub-crops of size 224x224 (corners + center with(out) horizontal flips).

Stochastic gradient descent with a mini-batch size of 128 was used to update the parameters, starting with a learning rate of 10^{-2}, in conjunction with a momentum term of 0,9. We anneal the learning rate throughout training manually when the validation error plateaus.

Dropout (Hinton et al., 2012) is used in the fully connected layers (6 and 7) with a rate of 0.5.

All weights are initialized to 10^{-2} and biases are set to 0.

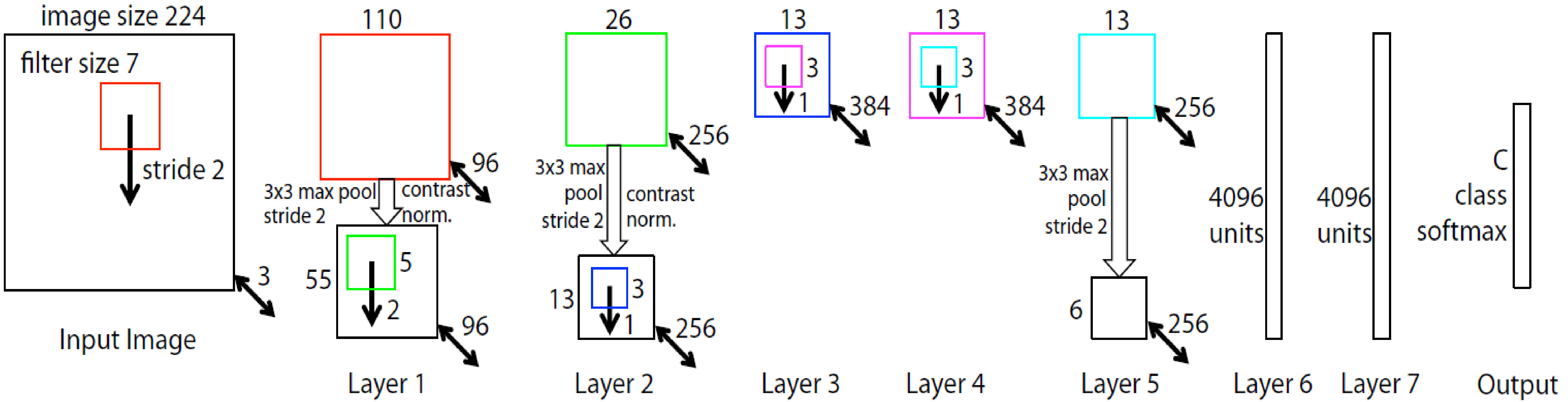
Visualization of the first layer filters during training reveals that a few of them dominate. To combat this, we renormalize each filter in the convolutional layers whose RMS value exceeds a fixed radius of 10^{-1} to this fixed radius. This is crucial, especially in the first layer of the model, where the input images are roughly in the [-128,128] range. As in (Krizhevsky et al., 2012), we produce multiple different crops and flips of each training example to boost training set size.

We stopped training after 70 epochs, which took around 12 days on a single GTX580 GPU, using an implementation based on (Krizhevsky et al., 2012).

Architecture Selection

By visualizing the first and second layers of Krizhevsky et al. 's architecture, various problems are apparent.

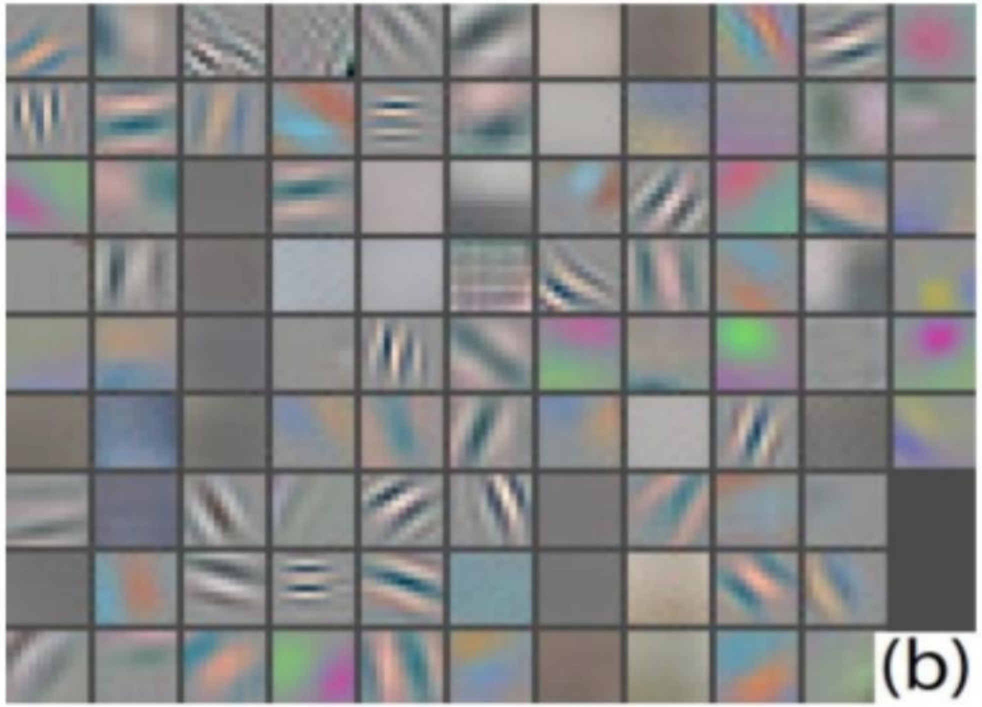
- ✓ Reduced the 1st layer filter size from 11x11 to 7x7;
- ✓ Made the stride of the convolution 2, rather than 4.



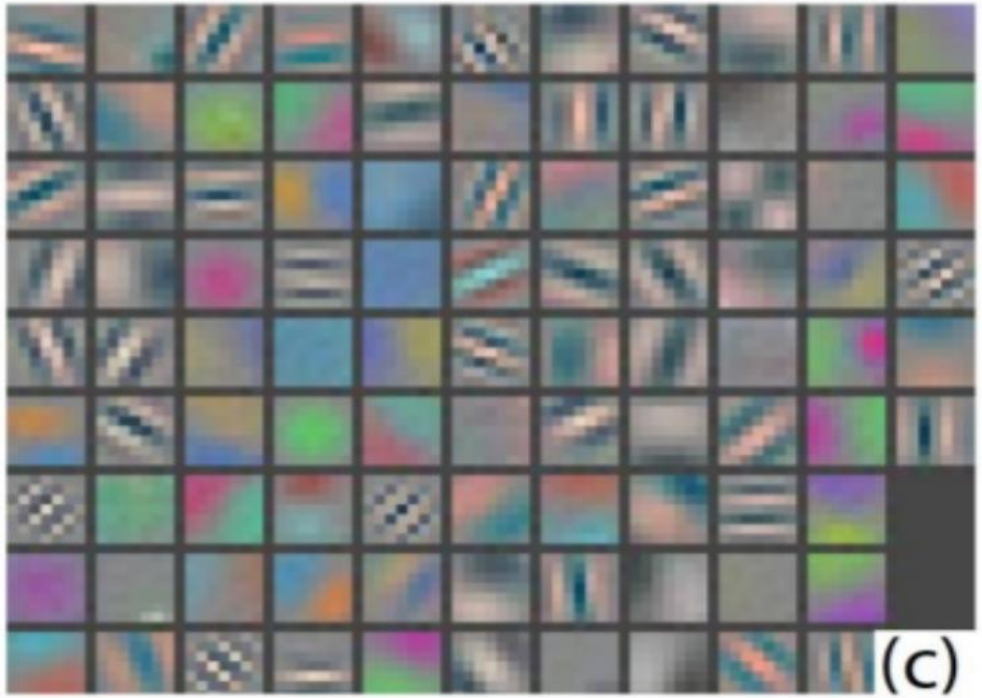
Architecture of our 8 layers convnet model. A 224 by 224 crop of an image (with 3 color planes) is presented as the input. This is convolved with 96 different 1st layer filters (red), each of size 7 by 7, using a stride of 2 in both x and y. The resulting feature maps are then: (i) passed through a rectified 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 6 256 = 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.

Zeiler-Fergus (ZF): Stride & filter size

The smaller stride (2 vs 4) and filter size (7x7 vs 11x11) results in more distinctive features and fewer “dead" features.



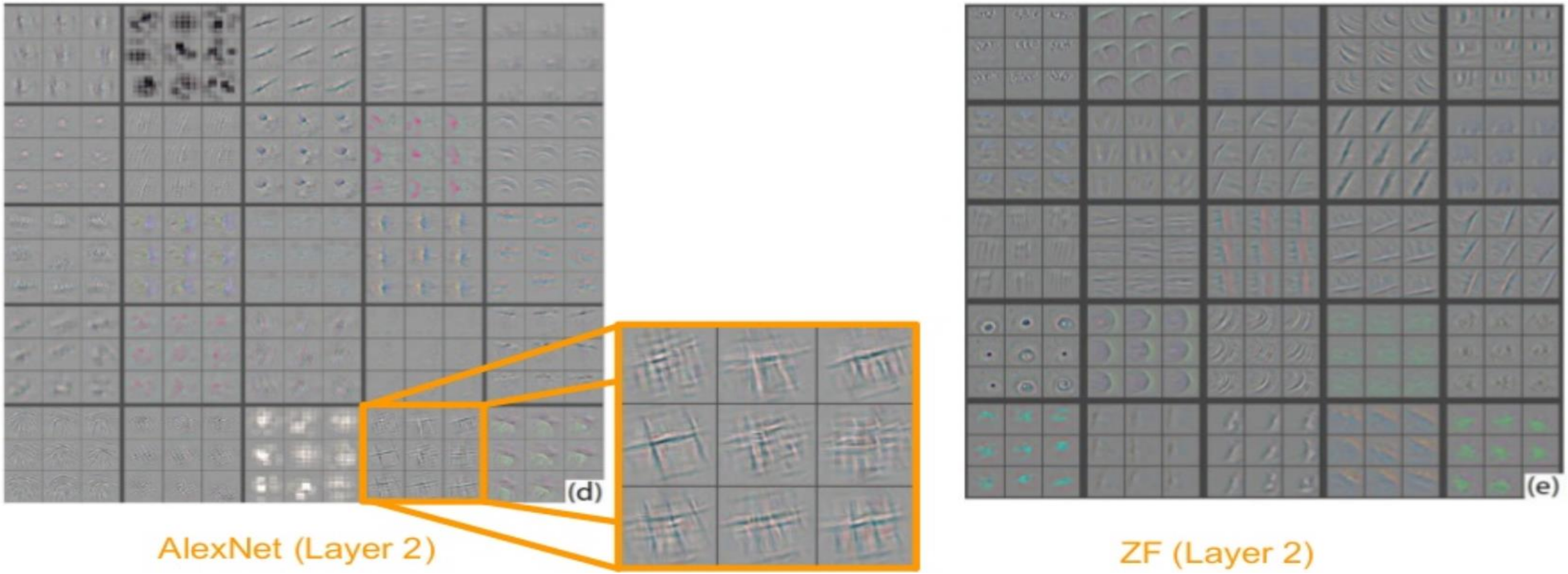
AlexNet (Layer1)



ZFNet (Layer1)

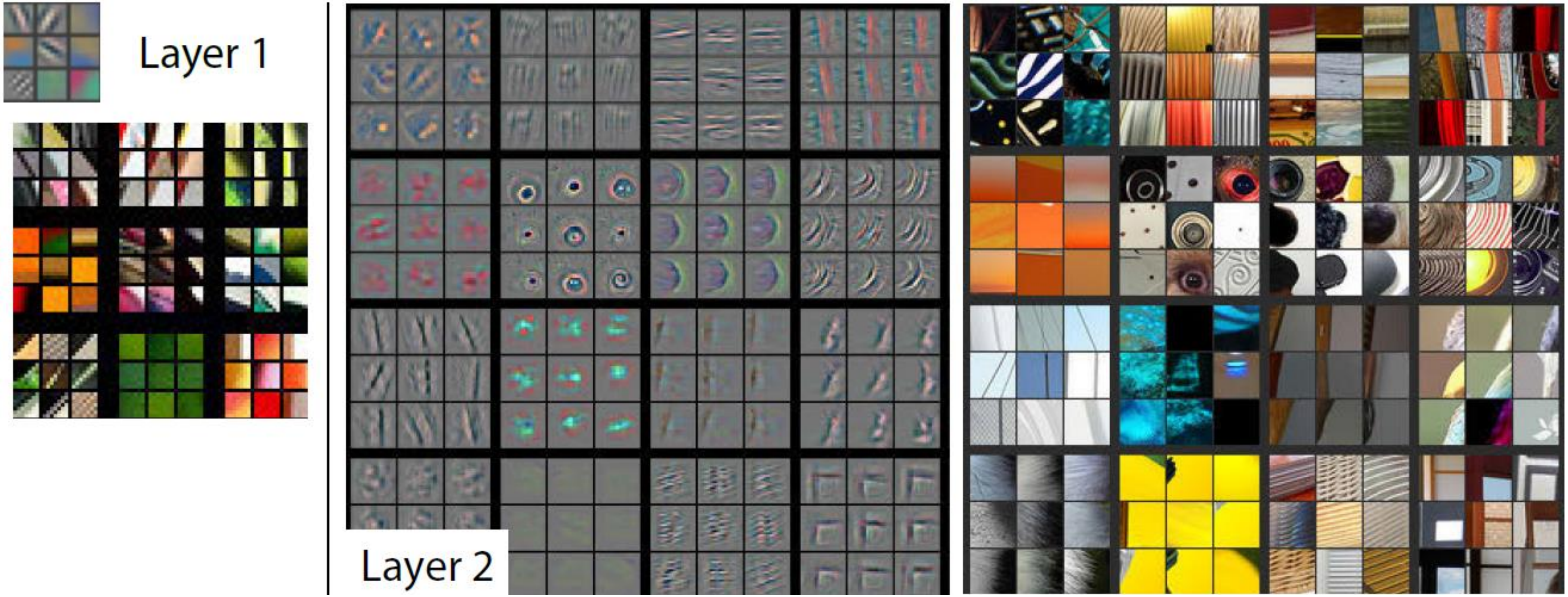
Zeiler-Fergus (ZF)

Cleaner features in ZF, without the aliasing artifacts caused by the stride 4 used in AlexNet.



Convnet Visualizations

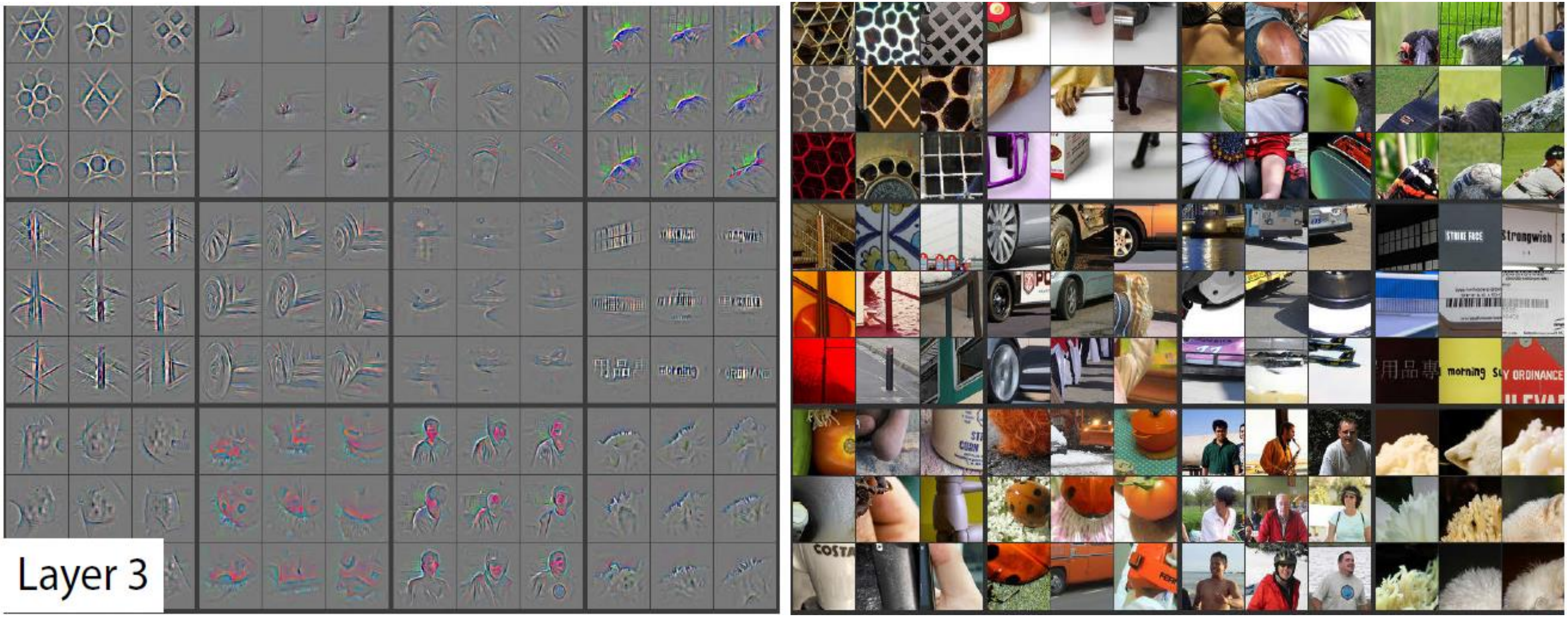
Use the deconvnet to visualize the feature activations on the ImageNet validation set.



Visualization of features in a fully trained model. 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.

Convnet Visualizations

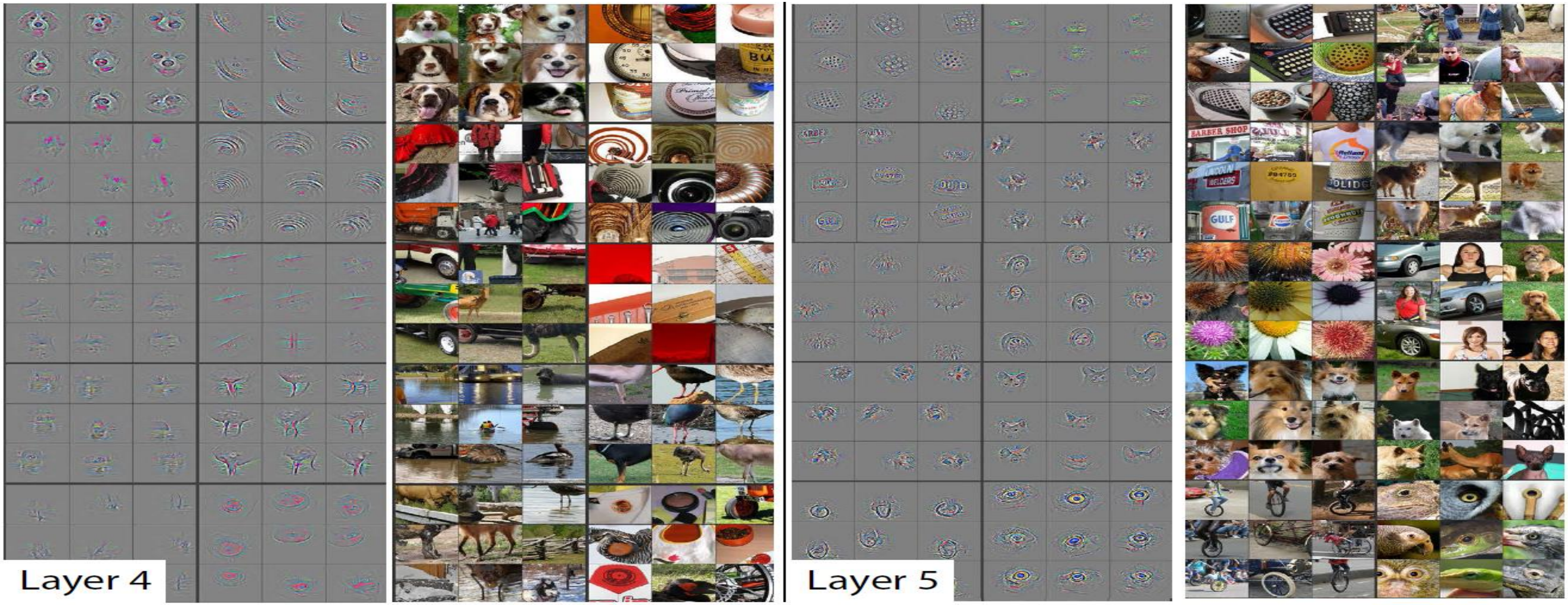
Use the deconvnet to visualize the feature activations on the ImageNet validation set.



Visualization of features in a fully trained model. 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.

Convnet Visualizations

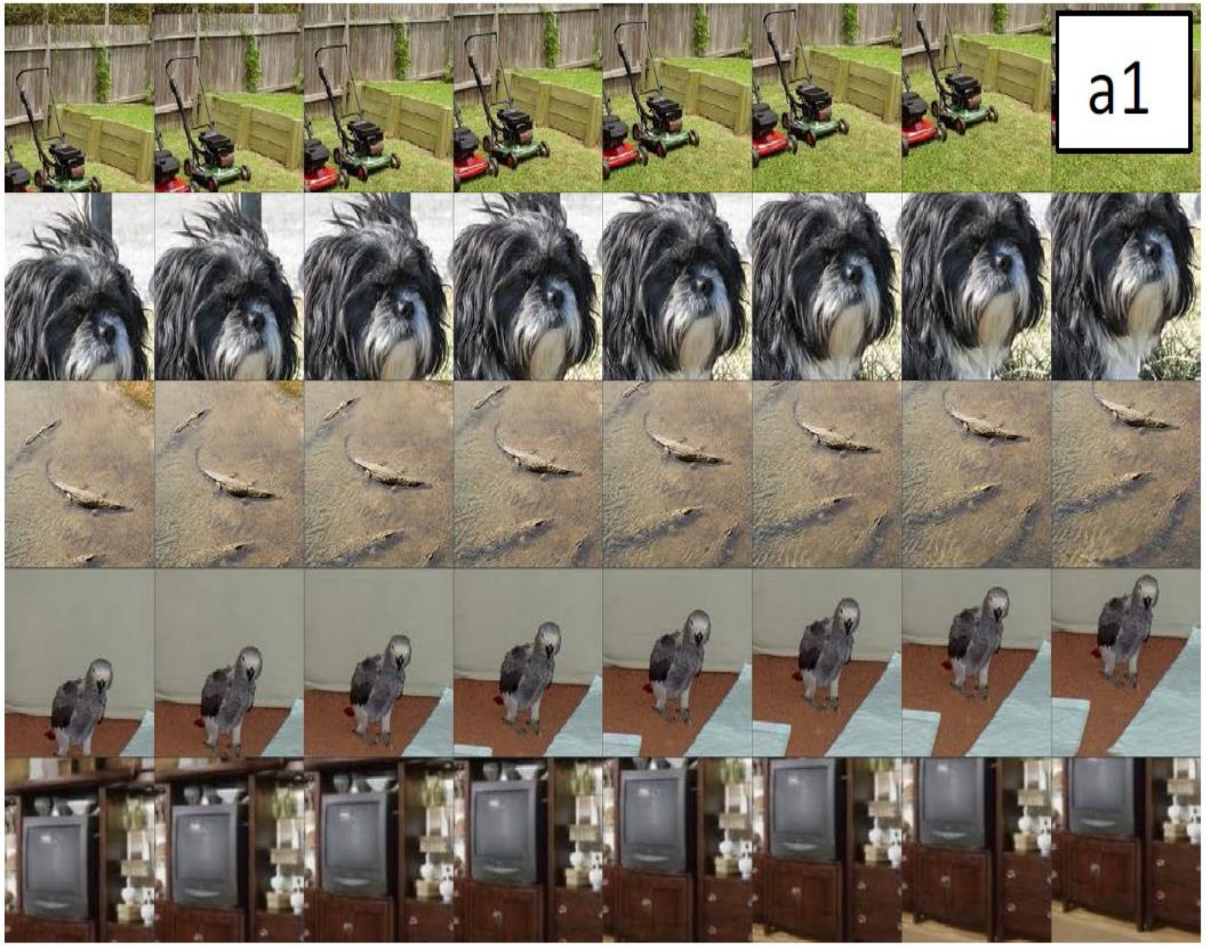
Use the deconvnet to visualize the feature activations on the ImageNet validation set.

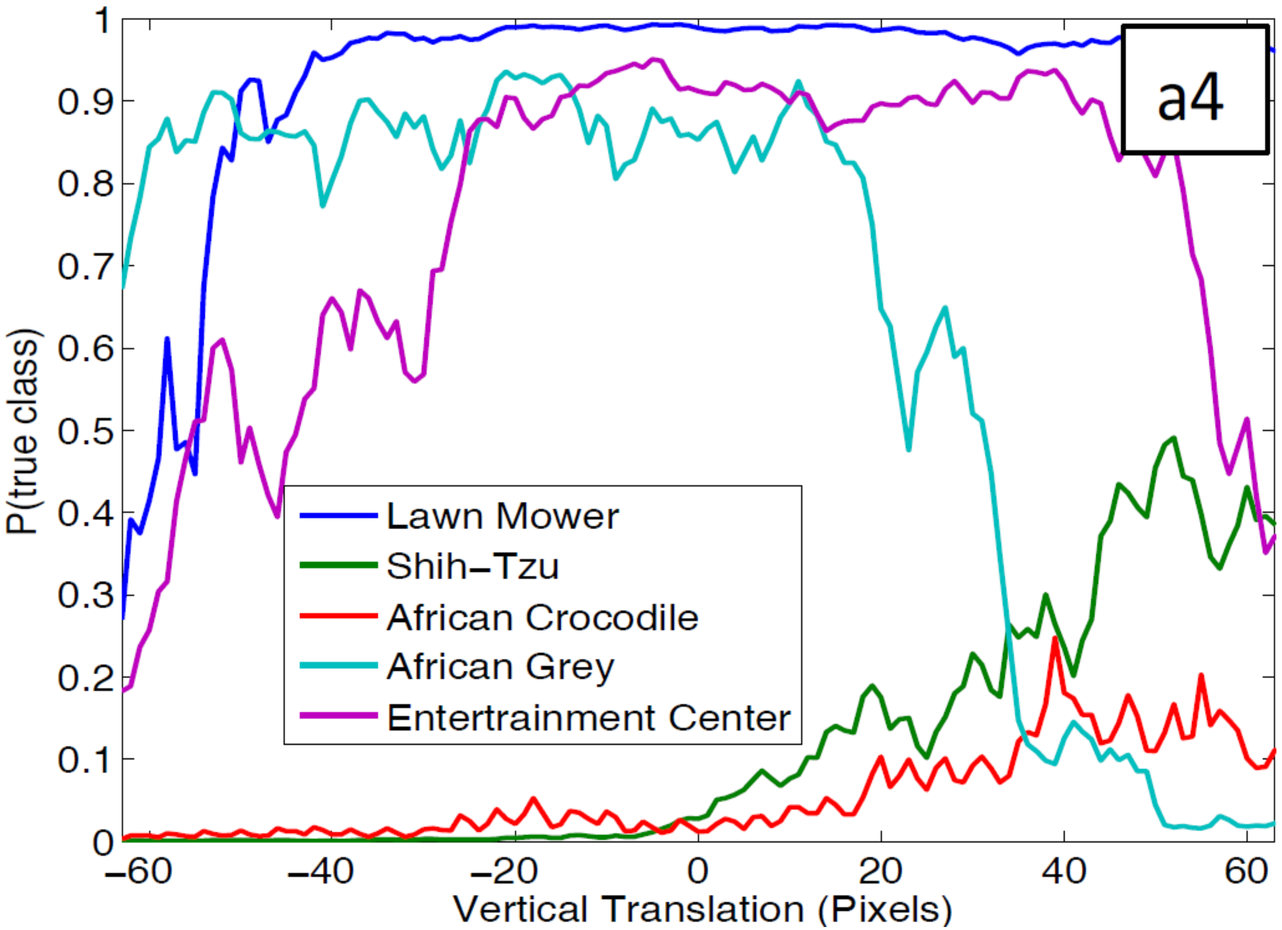


Visualization of features in a fully trained model. Show the top 9 activations in a random subset of feature maps across the validation data, projected down to pixel space using our deconvolutional network approach.

Convnet Visualizations

Output

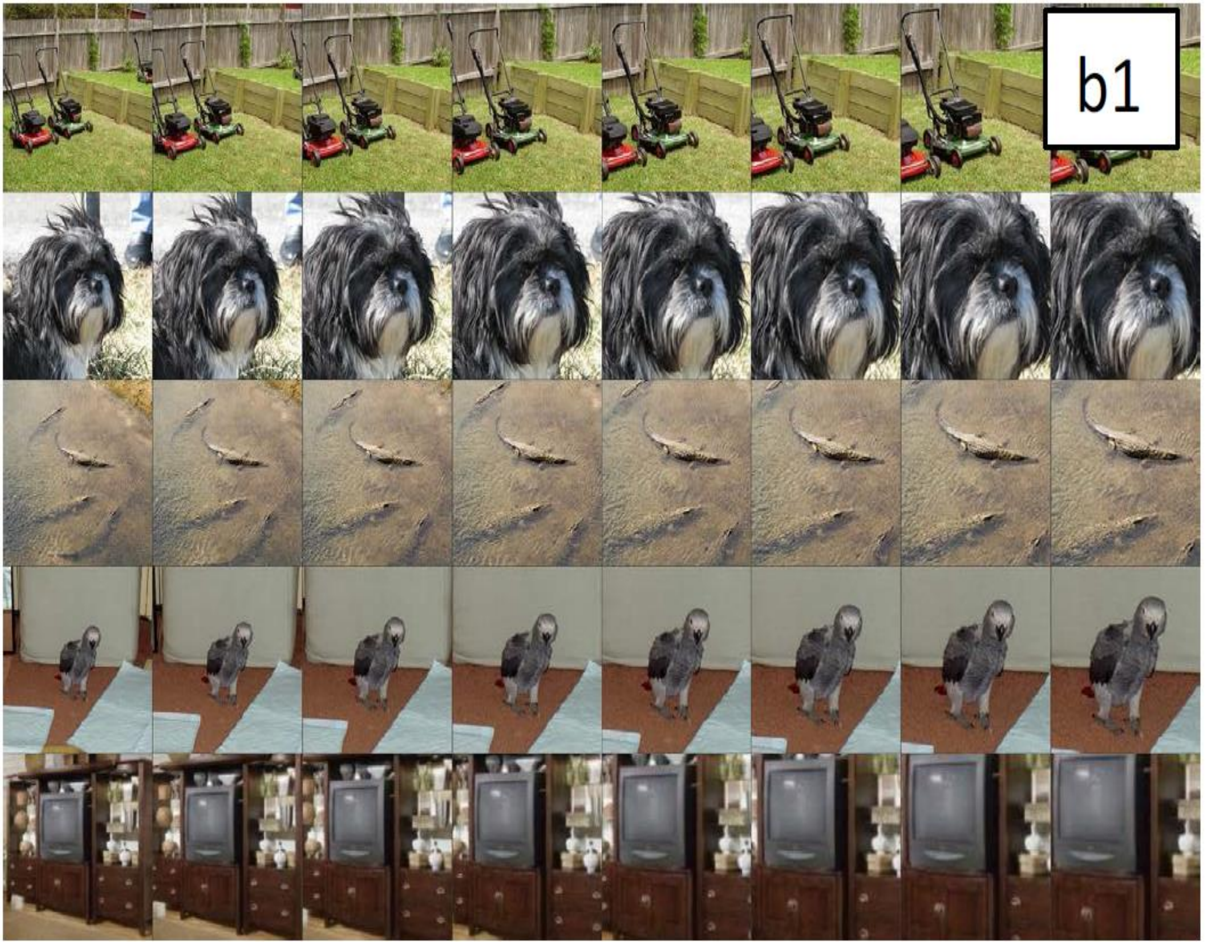


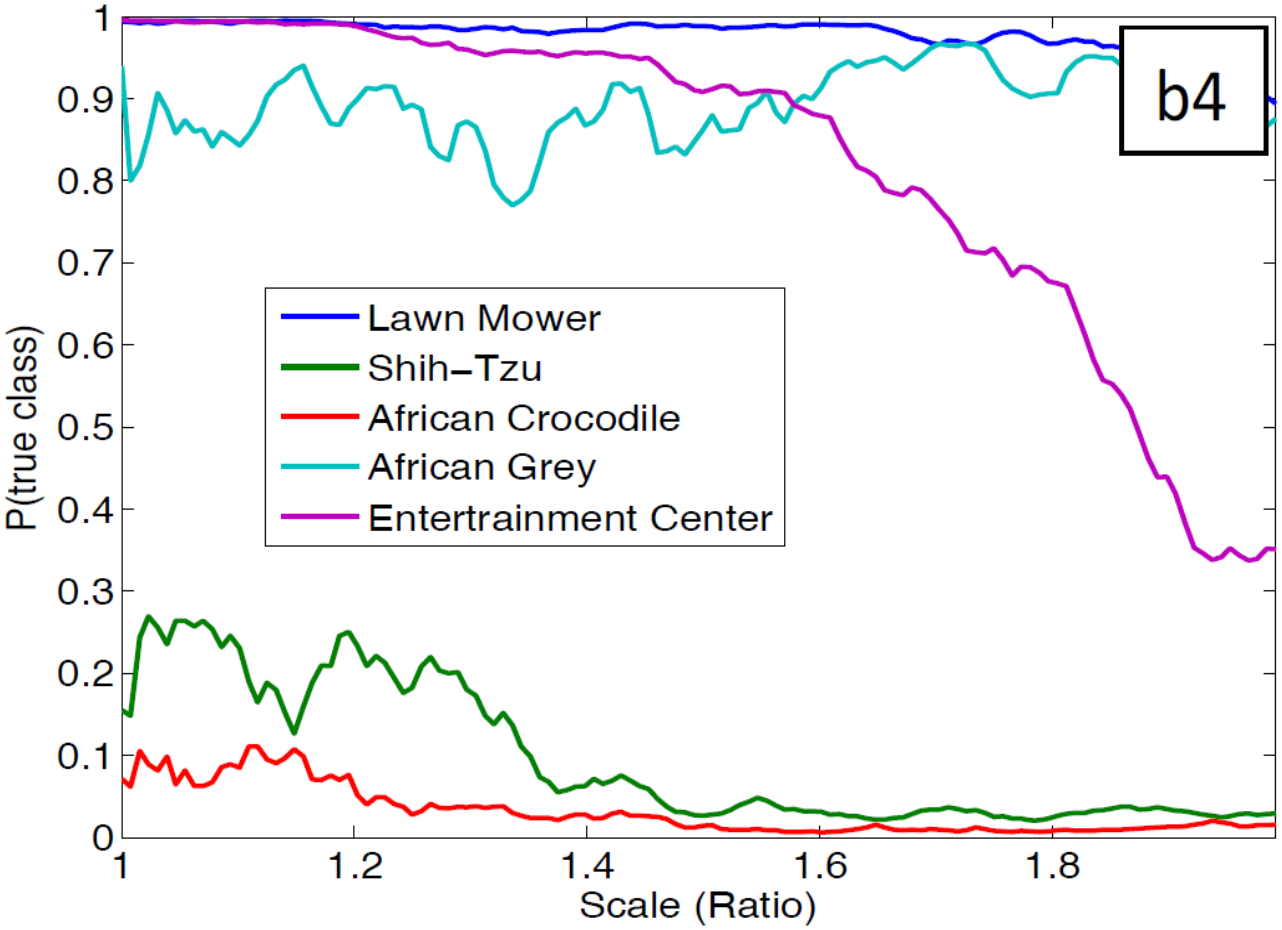


Analysis of vertical translation and the probability of the true label for each image, as the image is transformed.

Convnet Visualizations

Output

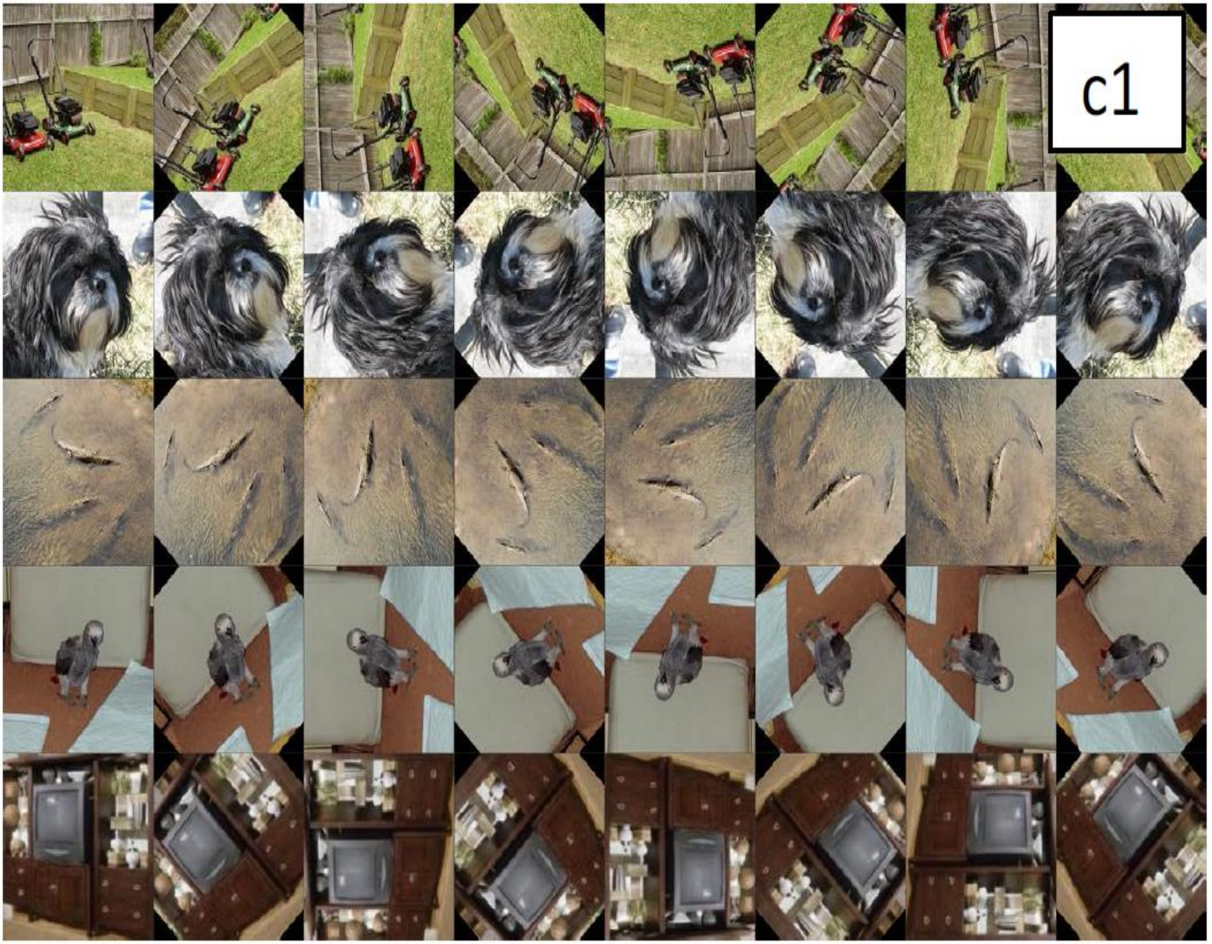


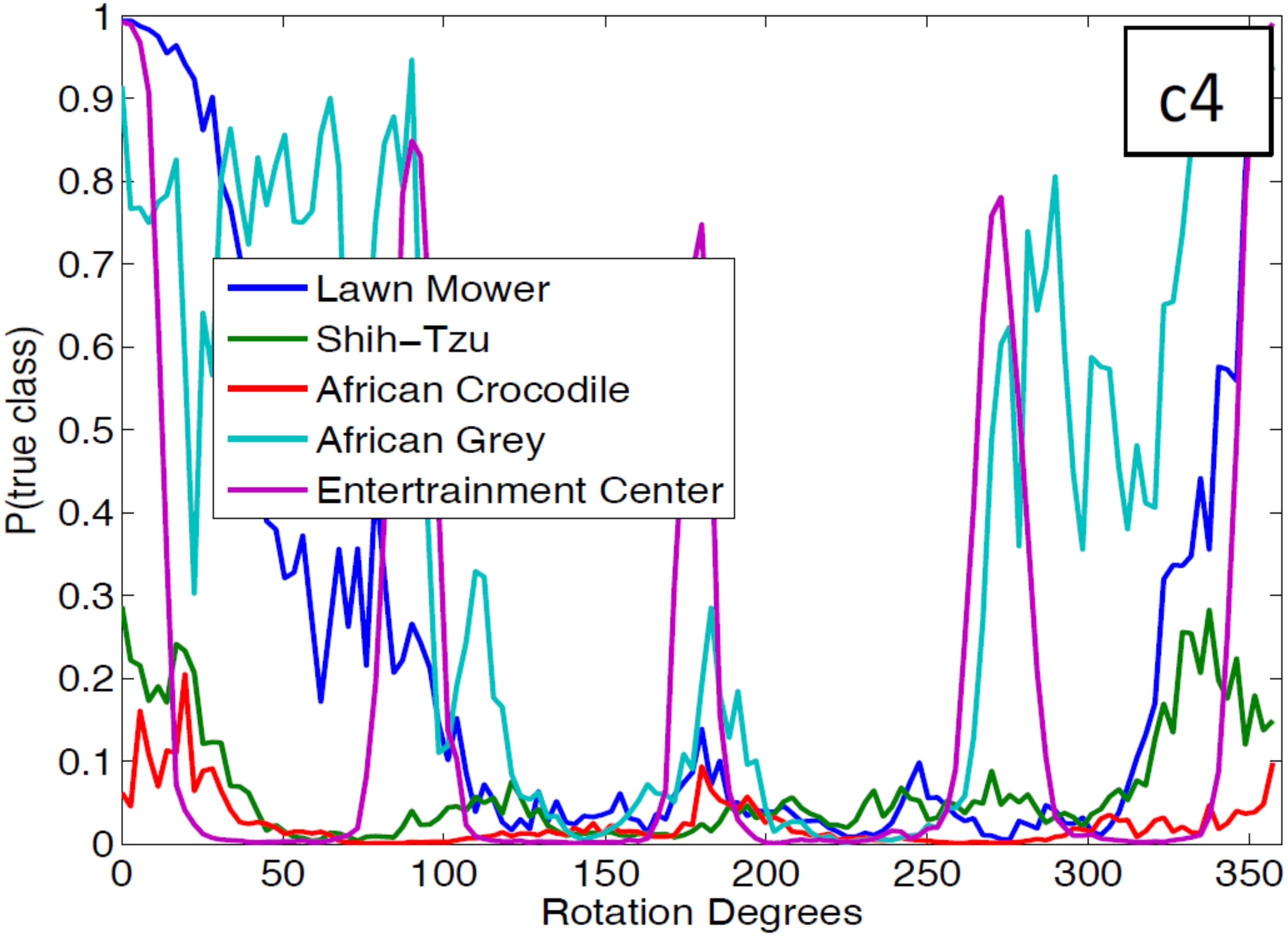


Analysis of scale and the probability of the true label for each image, as the image is transformed.

Convnet Visualizations

Output

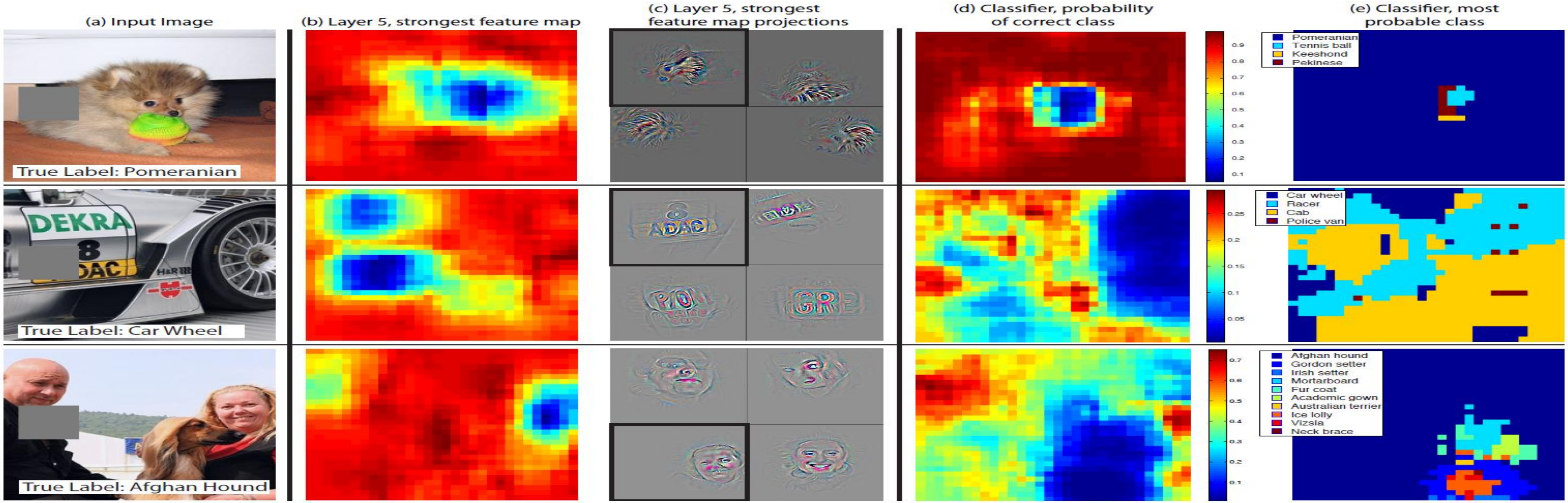




Analysis of rotation degrees and the probability of the true label for each image, as the image is transformed.

Convnet Visualizations

Use the deconvnet to visualize the feature activations on the ImageNet validation set.



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 classier output ((d) & (e)) changes. 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)). In the 2nd example, text on the car is the strongest feature in layer 5, but the classier is most sensitive to the wheel. The 3rd example contains multiple objects. The strongest feature in layer 5 picks out the faces, but the classier is sensitive to the dog (blue region in (d)), since it uses multiple feature maps.

Experiments

This dataset consists of 1.3M/50k/100k training/validation/test examples, spread over 1000 categories.

Using the exact architecture specified in (Krizhevsky et al., 2012), we attempt to replicate their result on the validation set.

We achieve an error rate within 0,1% of their reported value on the ImageNet 2012 validation set.

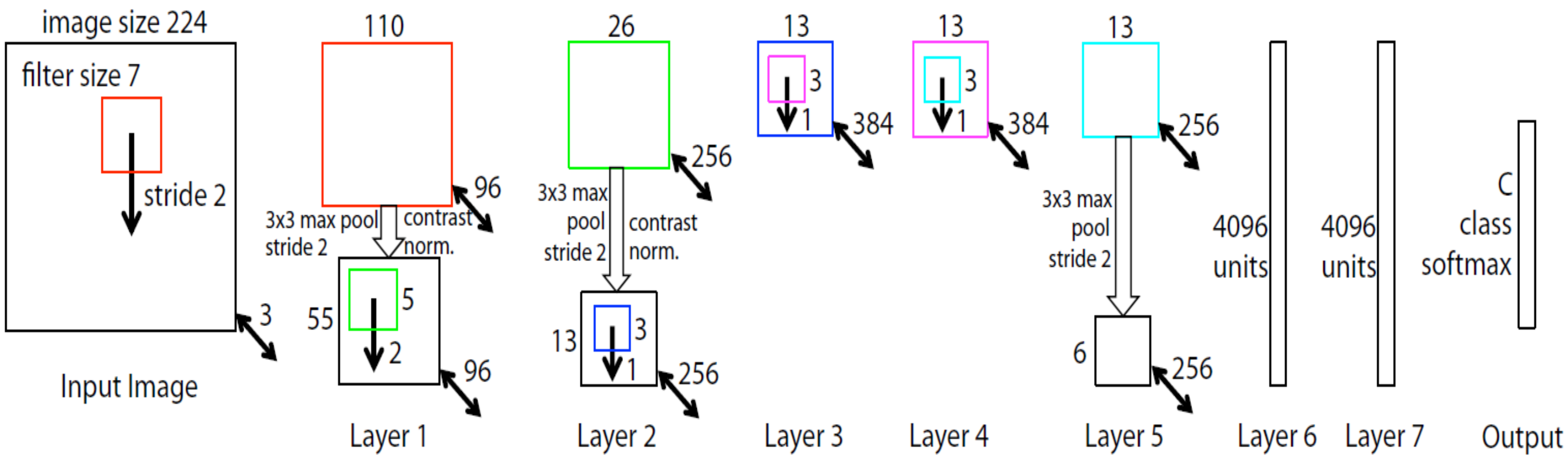
Next we analyze the performance of our model with the architectural changes (reduced the filter size from 11x11 to 7x7 in layer 1 and stride 2 rather than 4 of the convolutions in layers 1 & 2).

This model, shown, outperforms the architecture of (Krizhevsky et al., 2012), beating their single model result by 1,7% (test top-5). When we combine multiple models, we obtain a test error of 14,8%.

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

ImageNet 2012 classification error rates.

* indicates models that were trained on both ImageNet 2011 and 2012 training sets.



Experiments

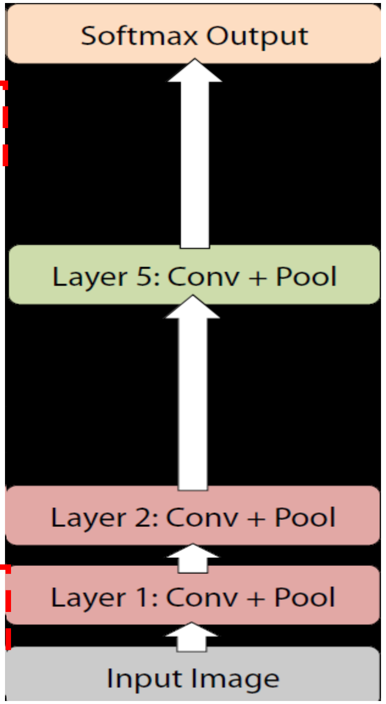
We first explore the architecture of (Krizhevsky et al., 2012) by adjusting the size of layers, or removing them entirely. In each case, the model is trained from scratch with the revised architecture.

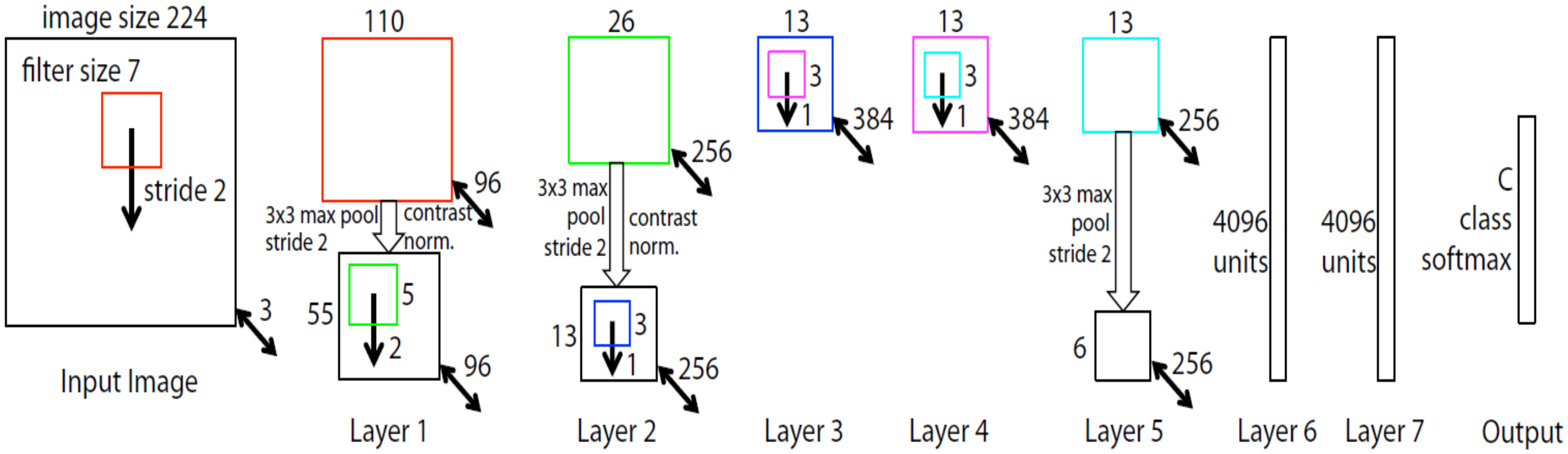
Removing the fully connected layers (6,7) only gives a slight increase in error. This is surprising, given that they contain the majority of model parameters.

Removing layers (3,4) also makes a relatively small different to the error rate. However, removing layers (3,4,6,7) the performance is dramatically worse.

We modify our model, changing the size of layers (6,7) makes little difference to performance. However, increasing the size of layers (3,4,5) give a useful gain in performance. But increasing these, while also enlarging the fully connected layers results in overfitting.

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



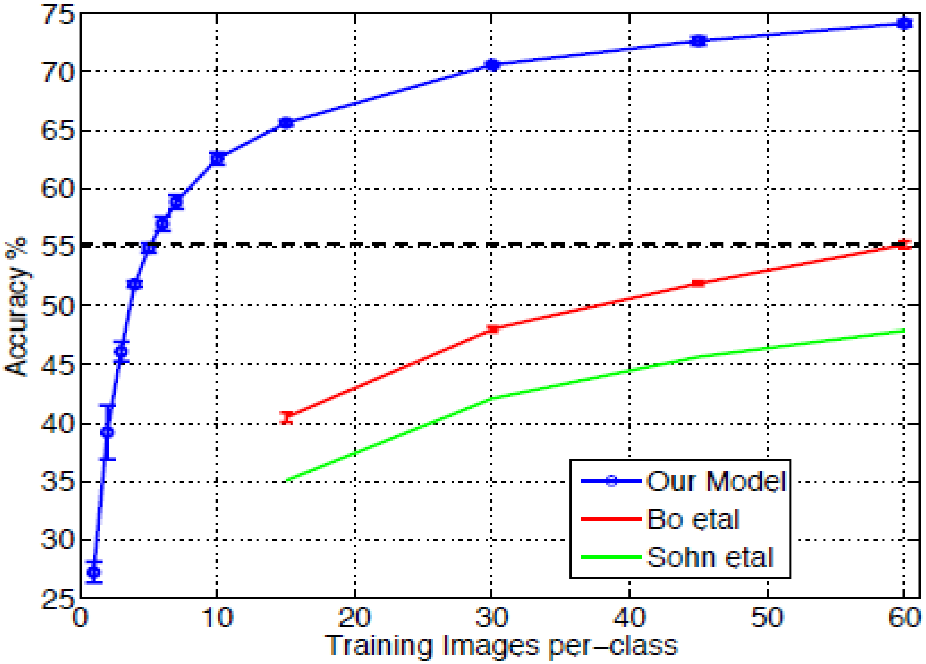


Experiments

The experiments above show the importance of the convolutional part of our ImageNet model in obtaining state-of-the-art performance.

We now explore the ability of these feature extraction layers to generalize to other datasets, namely Caltech-101 (Feifei et al., 2006), Caltech-256 (Griffin et al., 2006) and PASCAL VOC 2012.

To do this, we keep layers 1-7 of our ImageNet-trained model fixed and train a new softmax classifier on top (for the appropriate number of classes) using the training images of the new dataset.



# Train Cal-101	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	83.8 ± 0.5	86.5 ± 0.5

# Train Cal-256	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.	65.7 ± 0.2	70.6 ± 0.2	72.7 ± 0.4	74.2 ± 0.3

Pascal VOC 2012

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

Methods ([A]= (Sande et al., 2012) and [B] = (Yan et al., 2012)).

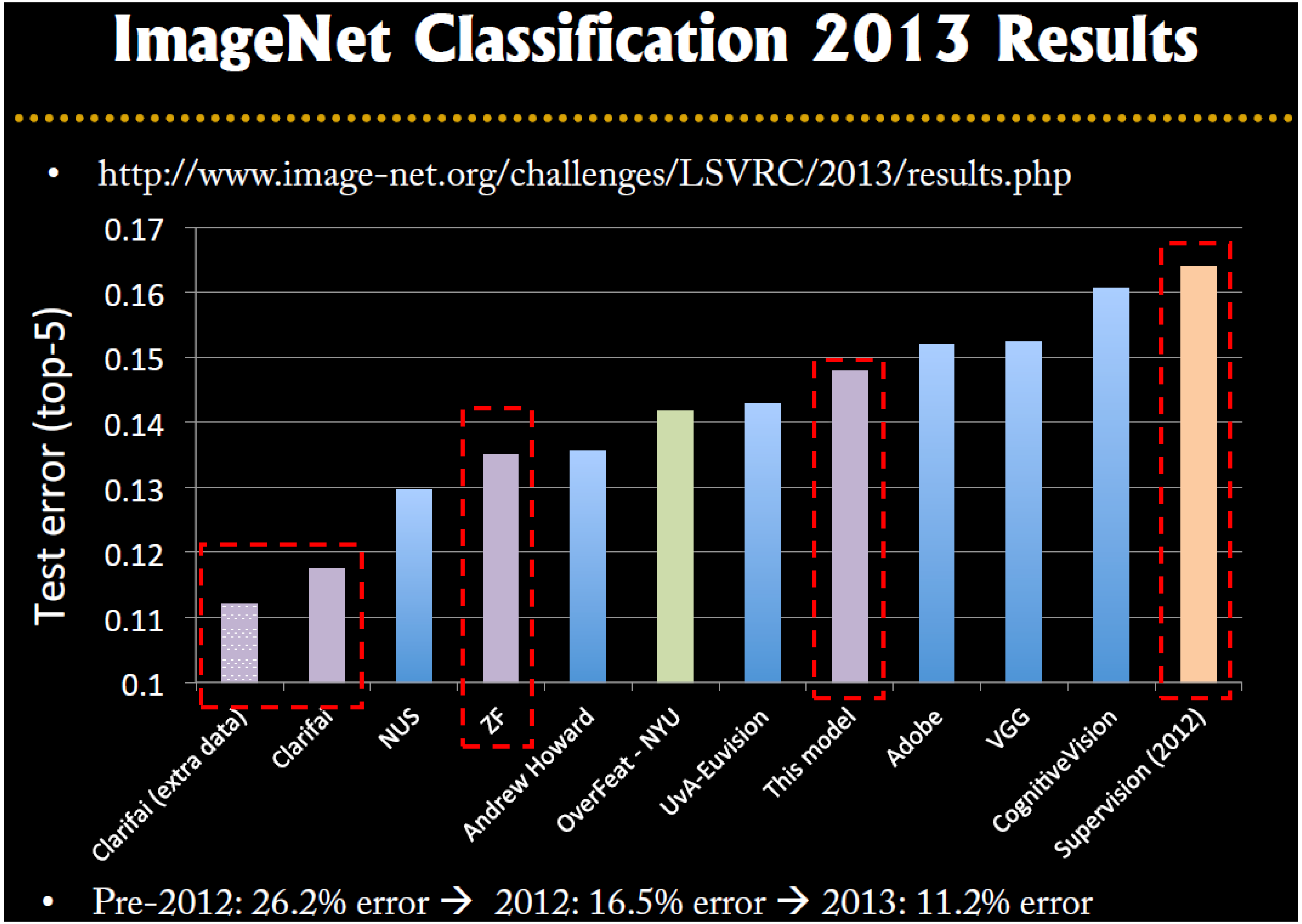
Discussions

We explored large convolutional neural network models, trained for image classification, in a number ways.

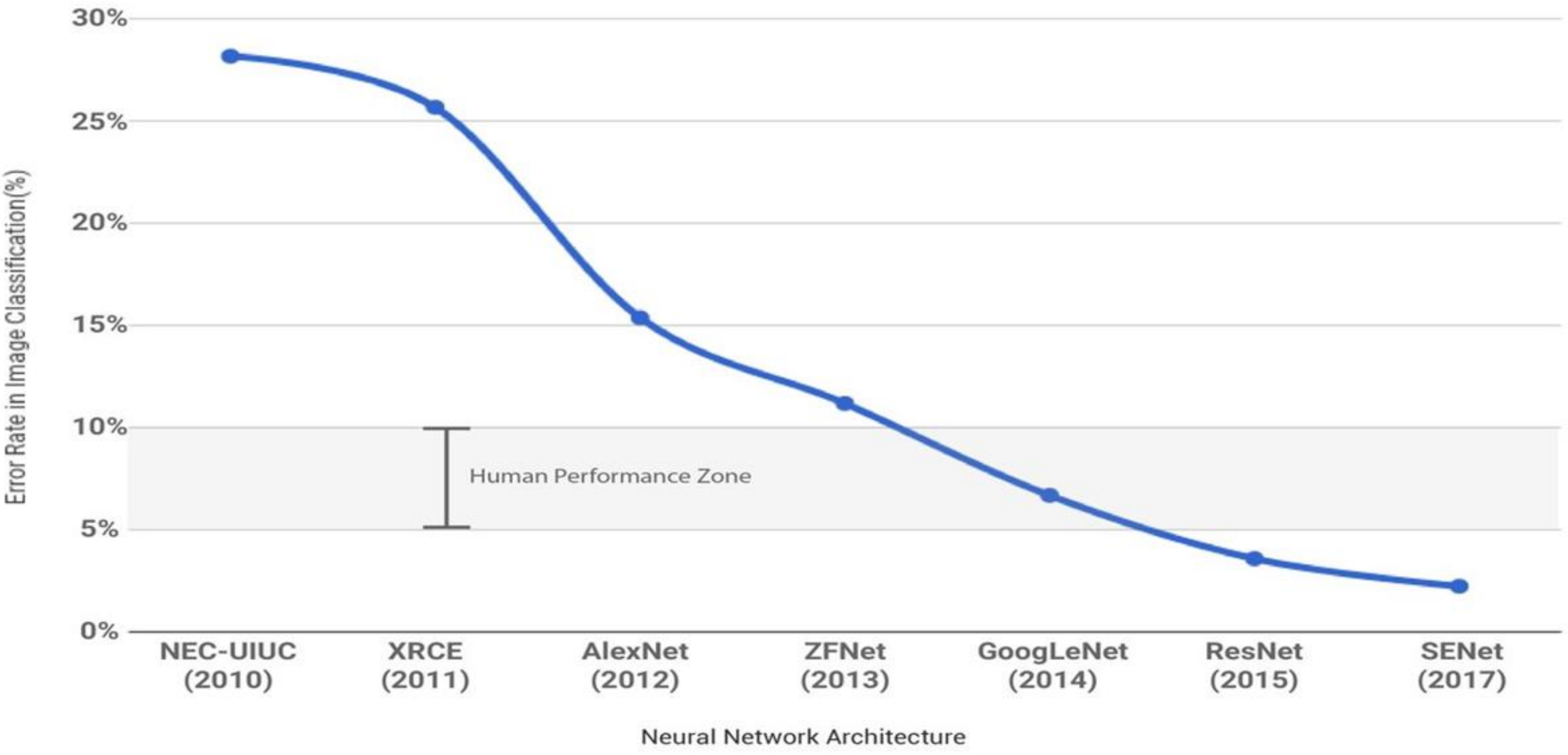
First, we presented a novel way to visualize the activity within the model.

We also showed how these visualization can be used to debug problems with the model to obtain better results, for example improving on Krizhevsky et al. 's (Krizhevsky et al., 2012) impressive ImageNet 2012 result.

We then demonstrated through a series of occlusion experiments that the model, while trained for classification, is highly sensitive to local structure in the image and is not just using broad scene context. An ablation study on the model revealed that having a minimum depth to the network, rather than any individual section, is vital to the model's performance.



Discussions



References

Hinton, G. E., Osindero, S., and The, Y. A fast learning algorithm for deep belief nets. Neural Computation, 18:1527{1554, 2006.

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

Krizhevsky, A., Sutskever, I., and Hinton, G.E. Imagenet classication with deep convolutional neural networks. In NIPS, 2012.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. Backpropagation applied to handwritten zip code recognition. Neural Comput., 1(4):541{551, 1989.

Sohn, K., Jung, D., Lee, H., and Hero III, A. Efficient learning of sparse, distributed, convolutional feature representations for object recognition. In ICCV, 2011.