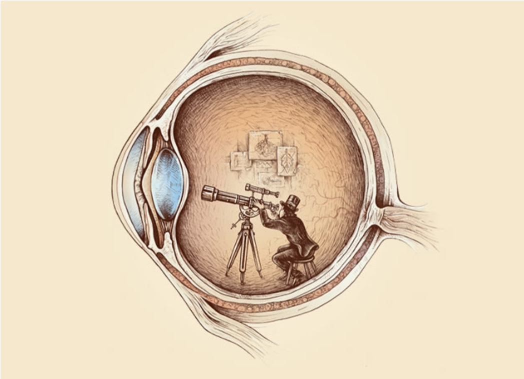


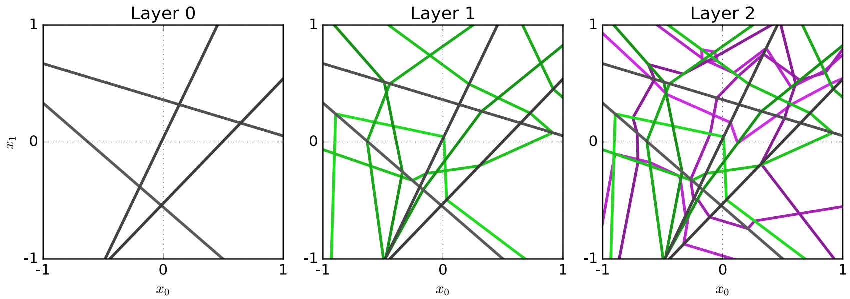
Computer Vision

CS 776 Fall 2018

Some Deep Networks for Recognition

Prof. Alex Berg

Structure of deep network outputs



Structure of deep network outputs

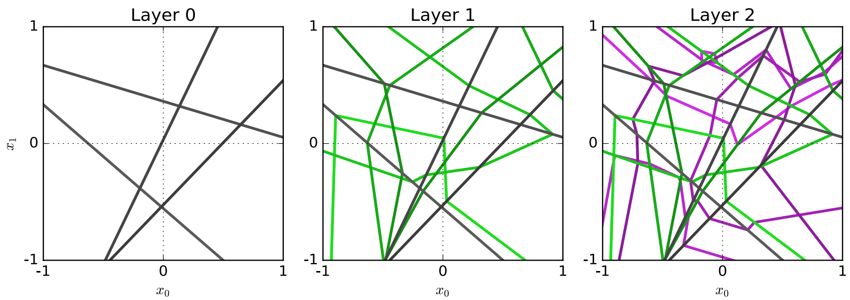


Figure 1. Deep networks with piecewise linear activations subdivide input space into convex polytopes. We take a three hidden layer ReLU network, with input $x \in \mathbb{R}^2$, and four units in each layer. The left pane shows activations for the first layer only. As the input is in $\mathbb{R}^2$, neurons in the first hidden layer have an associated line in $\mathbb{R}^2$, depicting their activation boundary. The left pane thus has four such lines. For the second hidden layer each neuron again has a line in input space corresponding to on/off, but this line is *different* for each region described by the first layer activation pattern. So in the centre pane, which shows activation boundary lines corresponding to second hidden layer neurons in green (and first hidden layer in black), we can see the green lines 'bend' at the boundaries. (The reason for this bending becomes apparent through the proof of Theorem 2.) Finally, the right pane adds the on/off boundaries for neurons in the third hidden layer, in purple. These lines can bend at both black and green boundaries, as the image shows. This final set of convex polytopes corresponds to all activation patterns for this network (with its current set of weights) over the unit square, with each polytope representing a different linear function.

On the Expressive Power of Deep Neural Networks Maithra Raghu Ben Poole Jon Kleinberg Surya Ganguli Jascha Sohl Dickstein (ICML 2017)

Structure of deep network outputs

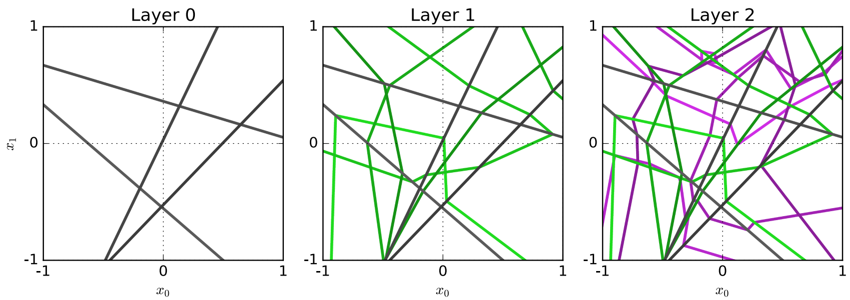


Figure 1. Deep networks with piecewise linear activations subdivide input space into convex polytopes. We take a three hidden layer ReLU network, with input $x \in \mathbb{R}^2$, and four units in each layer. The left pane shows activations for the first layer only. As the input is in $\mathbb{R}^2$, neurons in the first hidden layer have an associated line in $\mathbb{R}^2$, depicting their activation boundary. The left pane thus has four such lines. For the second hidden layer each neuron again has a line in input space corresponding to on/off, but this line is *different* for each region described by the first layer activation pattern. So in the centre pane, which shows activation boundary lines corresponding to second hidden layer neurons in green (and first hidden layer in black), we can see the green lines 'bend' at the boundaries. (The reason for this bending becomes apparent through the proof of Theorem 2.) Finally, the right pane adds the on/off boundaries for neurons in the third hidden layer, in purple. These lines can bend at both black and green boundaries, as the image shows. This final set of convex polytopes corresponds to all activation patterns for this network (with its current set of weights) over the unit square, with each polytope representing a different linear function.

On the Expressive Power of Deep Neural Networks Maithra Raghu Ben Poole Jon Kleinberg Surya Ganguli Jascha Sohl Dickstein (ICML 2017)

Structure of deep network outputs

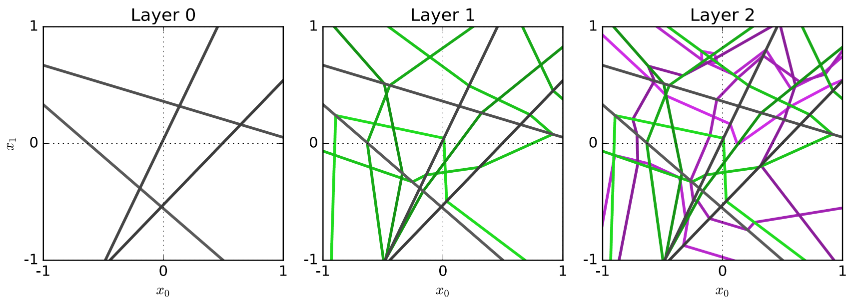
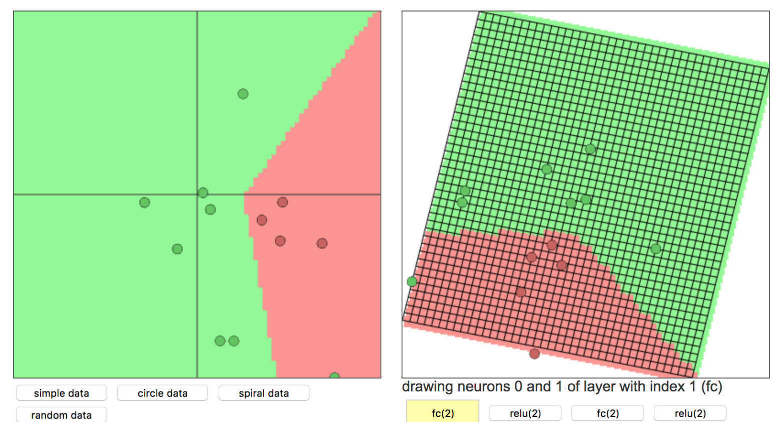


Figure 1. Deep networks with piecewise linear activations subdivide input space into convex polytopes. We take a three hidden layer ReLU network, with input $x \in \mathbb{R}^2$, and four units in each layer. The left pane shows activations for the first layer only. As the input is in $\mathbb{R}^2$, neurons in the first hidden layer have an associated line in $\mathbb{R}^2$, depicting their activation boundary. The left pane thus has four such lines. For the second hidden layer each neuron again has a line in input space corresponding to on/off, but this line is *different* for each region described by the first layer activation pattern. So in the centre pane, which shows activation boundary lines corresponding to second hidden layer neurons in green (and first hidden layer in black), we can see the green lines 'bend' at the boundaries. (The reason for this bending becomes apparent through the proof of Theorem 2.) Finally, the right pane adds the on/off boundaries for neurons in the third hidden layer, in purple. These lines can bend at both black and green boundaries, as the image shows. This final set of convex polytopes corresponds to all activation patterns for this network (with its current set of weights) over the unit square, with each polytope representing a different linear function.

On the Expressive Power of Deep Neural Networks Maithra Raghu Ben Poole Jon Kleinberg Surya Ganguli Jascha Sohl Dickstein (ICML 2017)

How does this compare to:



https://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html

Structure of deep networks

Structure of deep networks

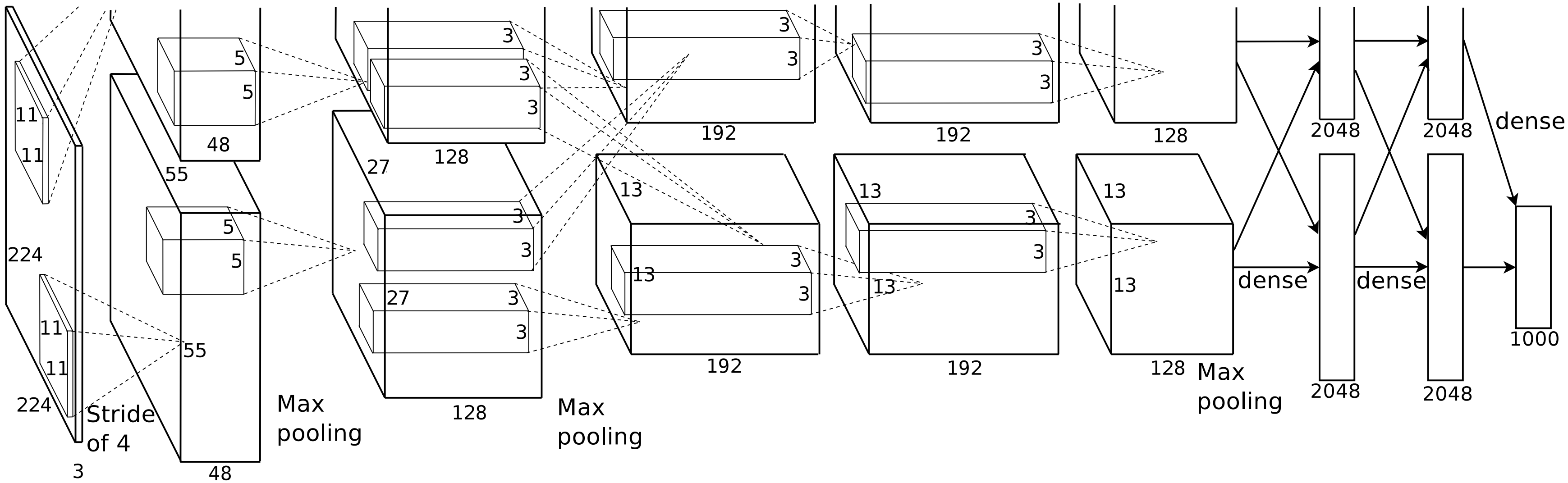


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky Ilya Sutskever Geoffrey E. Hinton (NIPS 2012)

Structure of deep networks

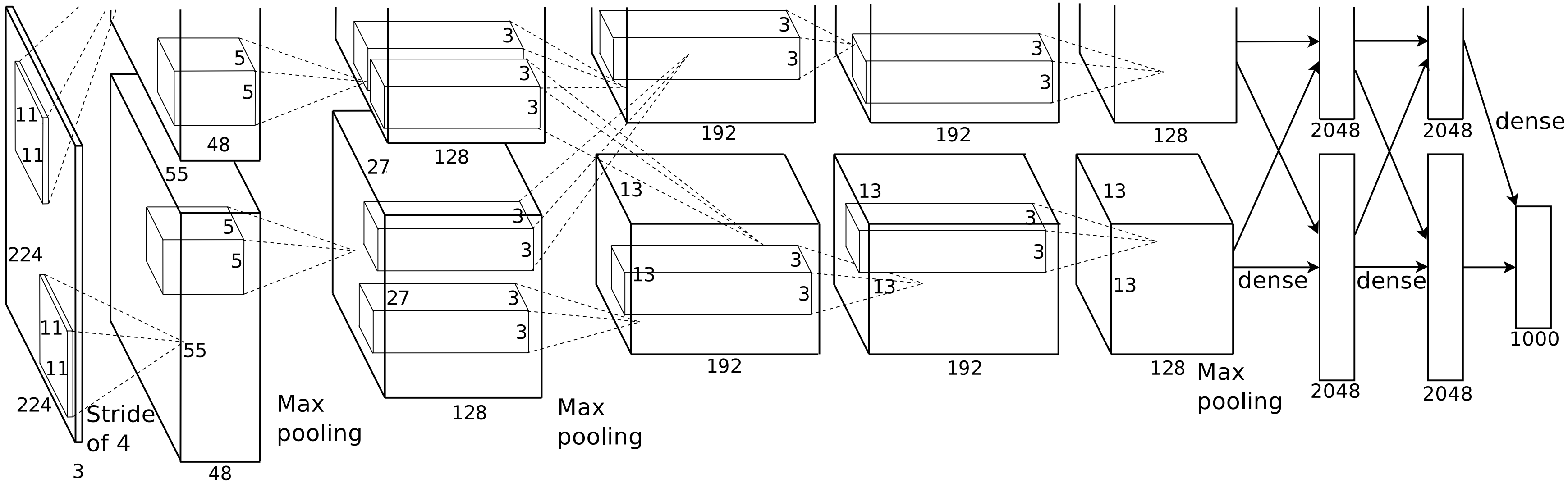
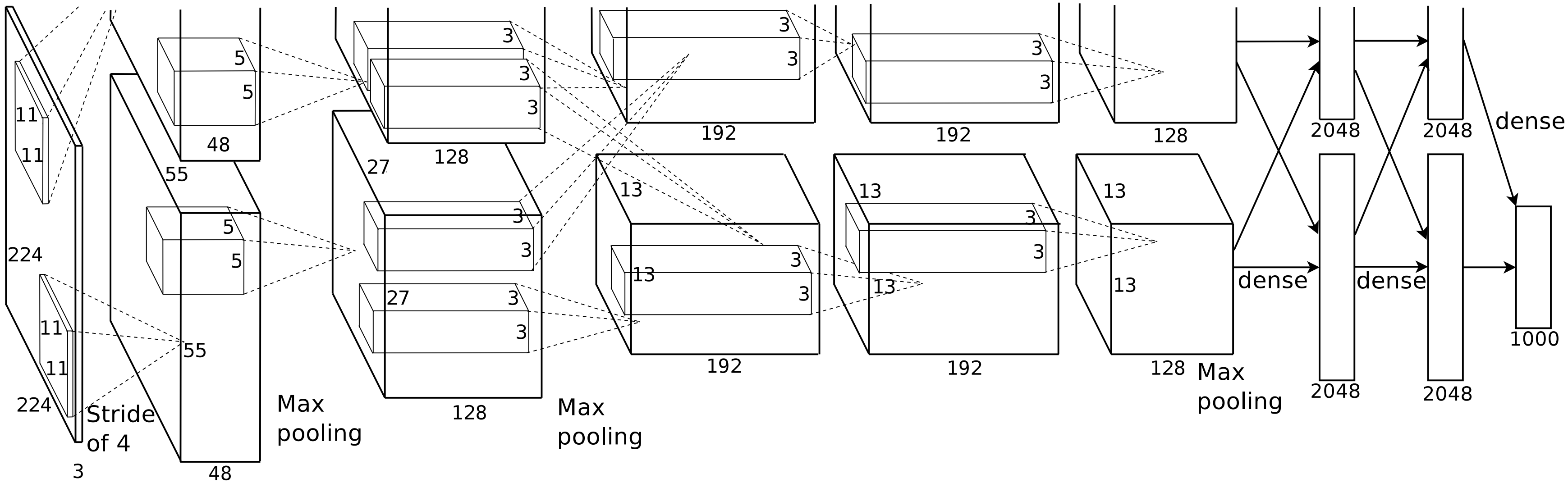
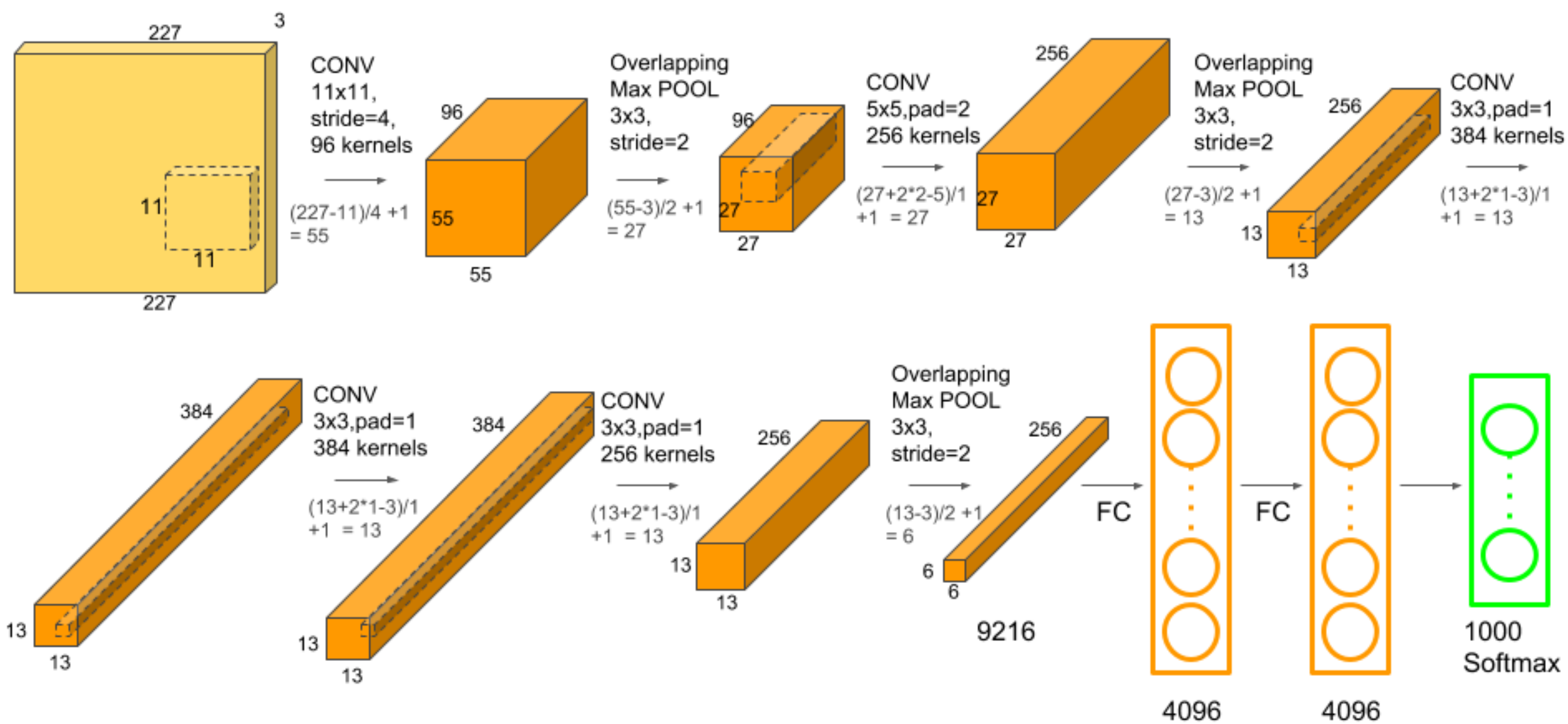


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

What is the loss function?
Where are the non-linearities?

Structure of deep networks





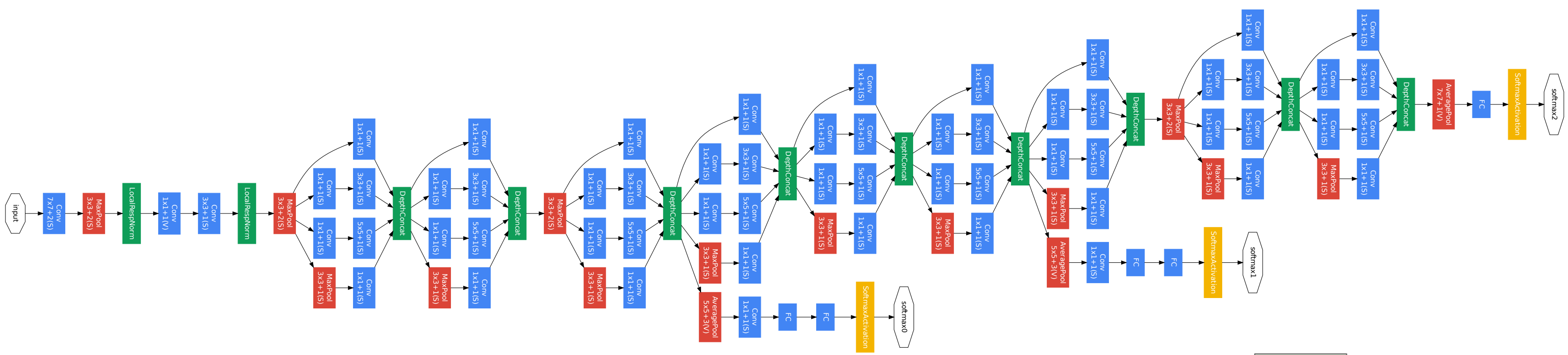
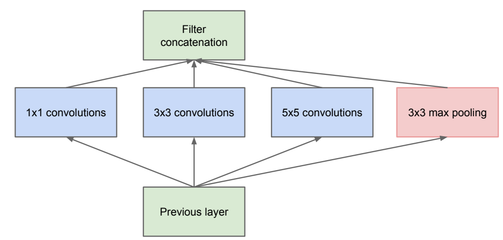
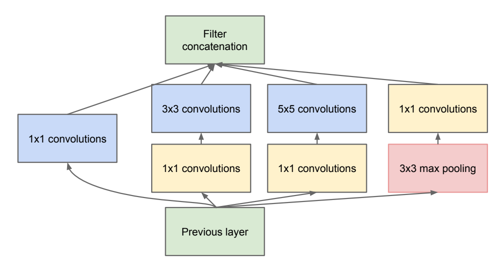


Figure 3: GoogLeNet network with all the bells and whistles.



(a) Inception module, naïve version



(b) Inception module with dimensionality reduction

Going Deeper with Convolutions

Christian Szegedy1, Wei Liu, …. Many others (CVPR 2015)

Going deeper!

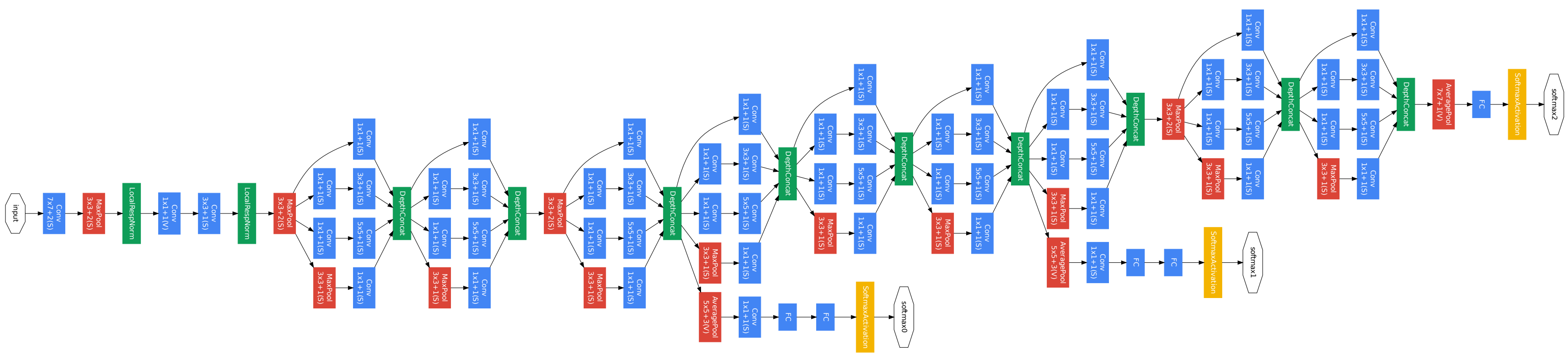


Figure 3: GoogLeNet network with all the bells and whistles.

Going Deeper with Convolutions

Christian Szegedy[1], Wei Liu, …. Many others (CVPR 2015)

VGG --- also deep (and cheap)

Table 1: **ConvNet configurations** (shown in columns). The depth of the configurations increases from the left (A) to the right (E), as more layers are added (the added layers are shown in bold). The convolutional layer parameters are denoted as "conv⟨receptive field size⟩-⟨number of channels⟩". The ReLU activation function is not shown for brevity.

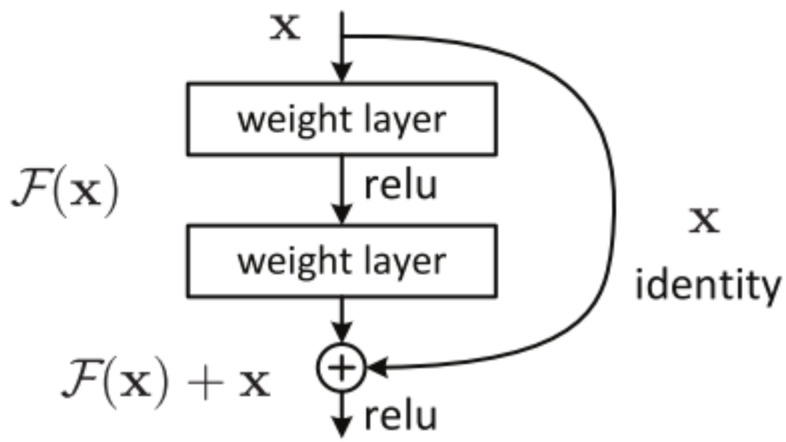
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224×224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	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

VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION Karen Simonyan ∗ & Andrew Zisserman + (ICLR 2015)

Cleaning things up: Resnet



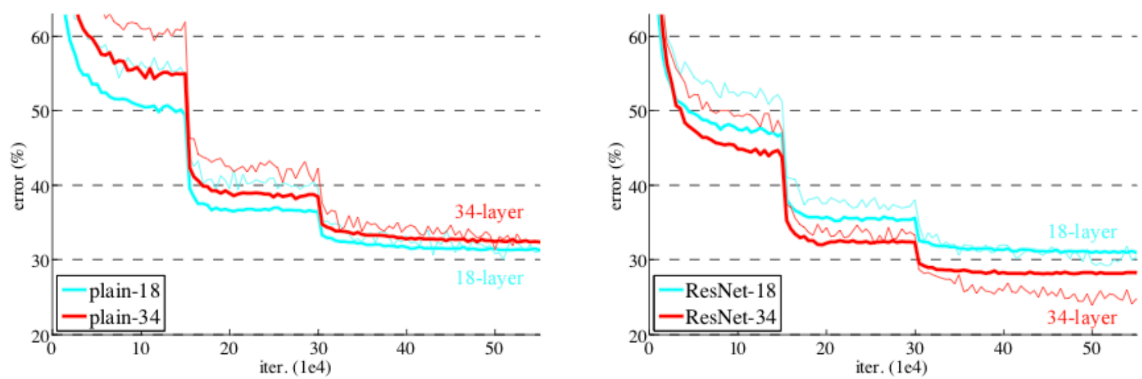
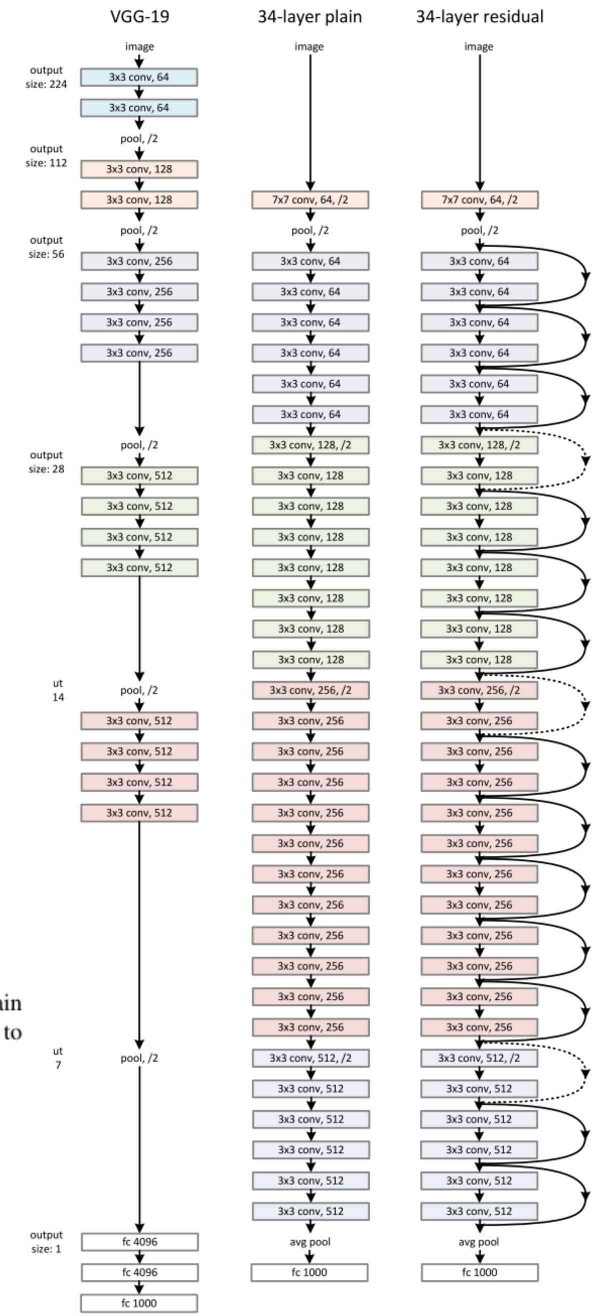


Figure 4. Training on **ImageNet**. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.



Deep Residual Learning for Image Recognition

Kaiming He Xiangyu Zhang Shaoqing Ren, Jian Sun

(CVPR 2016)