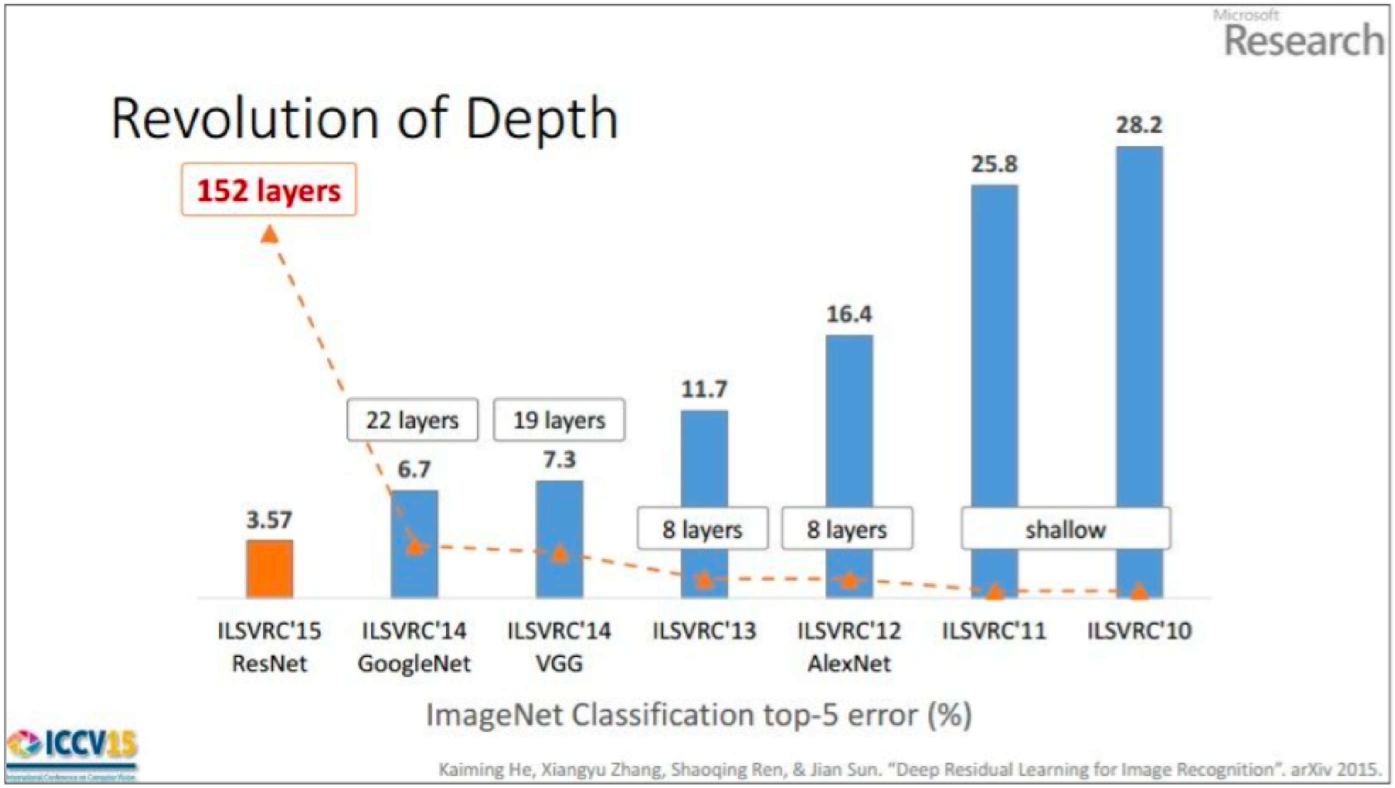
TTIC 31230, Fundamentals of Deep Learning

David McAllester, Winter 2018

Convolutional Neural Networks — CNNs

CNNs

Imagenet Classification. 1000 kinds of objects.

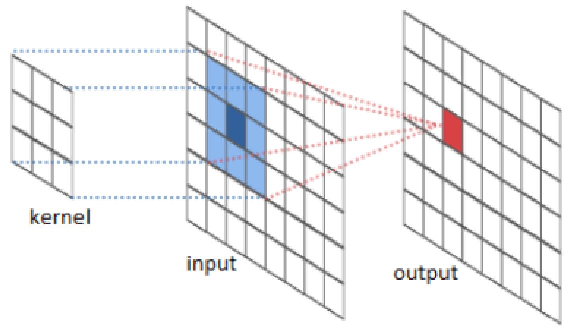


(slide from Kaiming He's recent presentation)

2016 error rate is 3.0%   2017 error rate is 2.25%

A CNN

A convolution slides a filter (a kernel) across an image.



$$W \qquad\qquad x \qquad\qquad y$$

River Trail Documentation

$$y[b,i,j,c_y] \;=\; W[\Delta i, \Delta j, c_x, c_y] x[b, i+\Delta i, j+\Delta j, c_x]$$

$$y[b,i,j,c_y] \text{ += } B[c_y]$$

Use Swap Rule to get Backward Method

$$y[b,i,j,c_y] \;=\; W[\Delta i,\Delta j,c_x,c_y]x[b,i+\Delta i,j+\Delta j,c_x]$$

$$W.\mathrm{grad}[\Delta i,\Delta j,c_x,c_y] \mathrel{+}= y.\mathrm{grad}[b,i,j,c_y]x[b,i+\Delta i,j+\Delta_j,c_x]$$

$$x.\mathrm{grad}[b,i{+}\Delta i,j{+}\Delta j,c_x] \mathrel{+}= W[\Delta i,\Delta j,c_x,c_y]y.\mathrm{grad}[b,i,j,c_y]$$

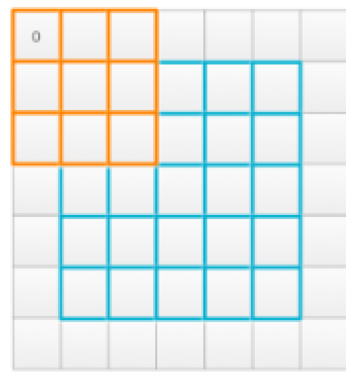
A Convolution Class in EDF

In EDF we would define a class for convolution parameter packages.

We would then construct the computation node for the output using Python code.

```
Y = Relu(Conv(Phi,X)).
```

Padding



Jonathan Hui

If we pad the input with zeros then the input and output can have the same spatial dimensions.

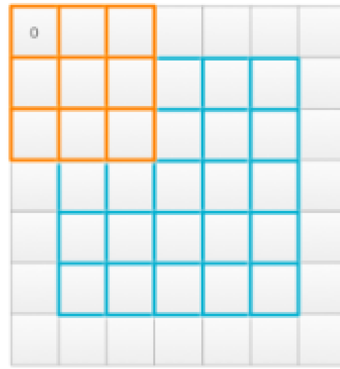
Zero Padding in NumPy

In NumPy we can add a zero padding of width p to an image as follows:

```
padded = np.zeros(W + 2*p,  H + 2*p)

padded[p:W+p, p:H+p] = x
```

Padding



Jonathan Hui

Let x' (full square) be the padding of x (blue square). y also has the blue shape.

$$y[b,i,j,c_y] = W[\Delta i, \Delta j, c_x, c_y]x'[b, i+\Delta i, j+\Delta j, c_x] + B[c_y]$$

For padding p and a filter of width $2p+1$, we get that y has the same spatial dimensions as x.

Strides

We can move the filter by a “stride” s for each spatial step.

$$y[b,i,j,c_y] = W[\Delta i, \Delta j, c_x, c_y]x[b, s*i+\Delta i, s*j+\Delta j, c_x] + B[c_y]$$

Max Pooling

$$y[b,i,j,c] = \max_{\Delta i, \Delta j} \; x[b, s*i+\Delta i, \; s*j+\Delta j, \; c]$$

This is typically done with a stride greater than one so that the image dimension is reduced.

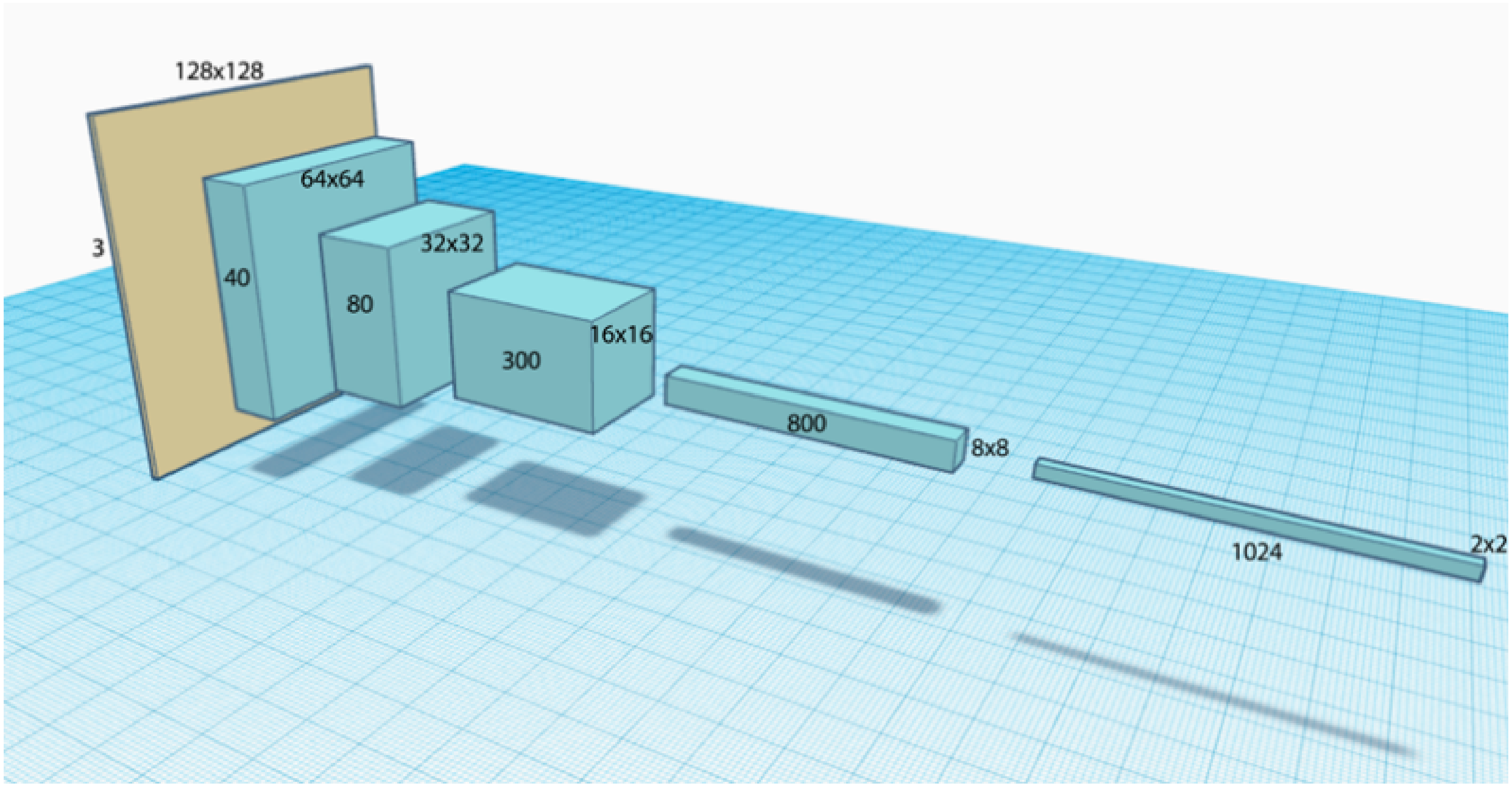
Fully Connected (FC) Layers

We reshape $x[b, x, y, c]$ to $x[b, (x, y, c)]$ and convert to using an MLP.

Basics

- Padding
- Convolution
- Stides
- Max Pooling
- Fully Connected Layers

A Sequence of "Images"



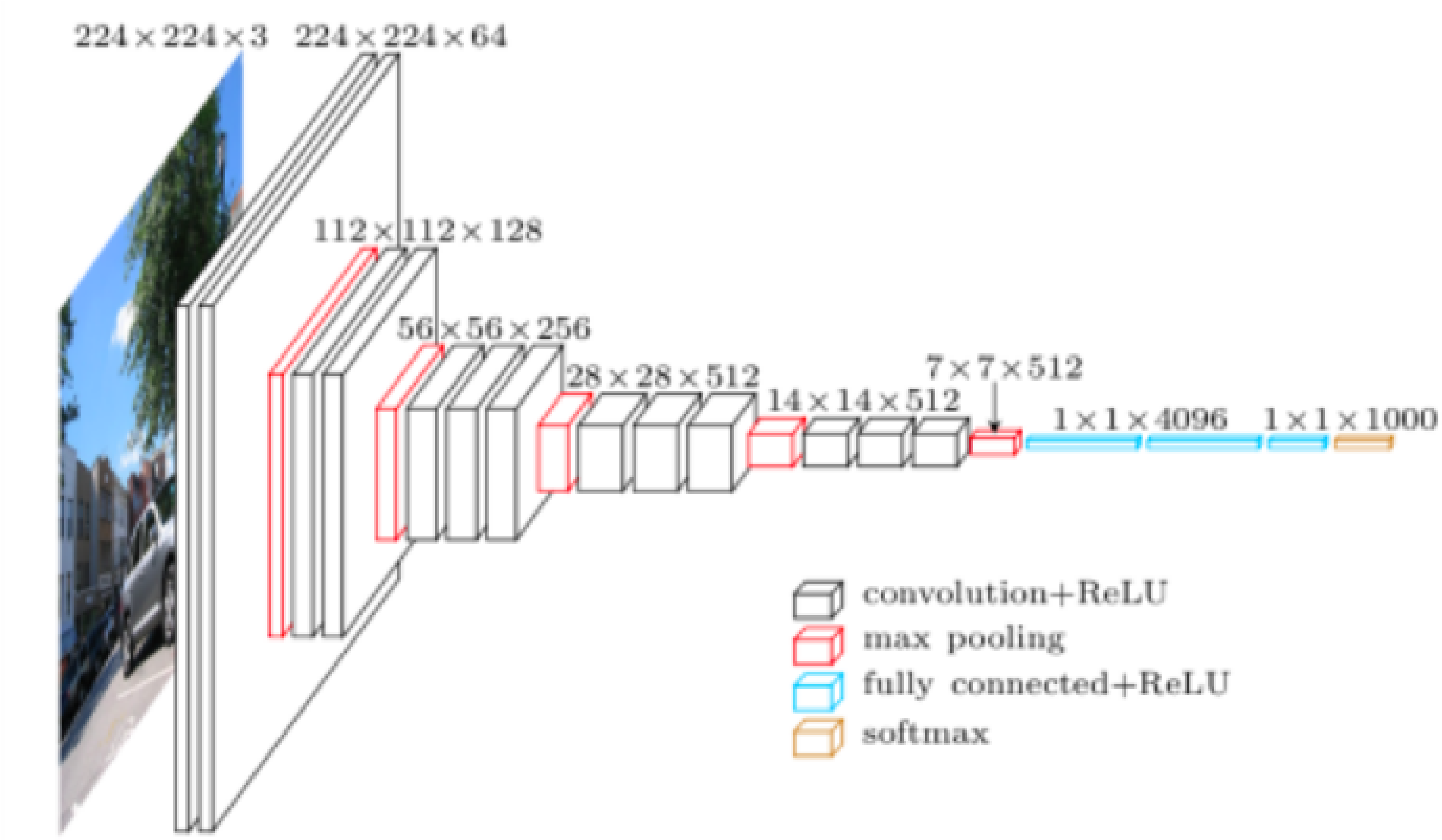
Jonathan Hui

Alexnet

Full (simplified) AlexNet architecture:
[227x227x3] INPUT
[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
[27x27x96] MAX POOL1: 3x3 filters at stride 2
[27x27x96] NORM1: Normalization layer
[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
[13x13x256] MAX POOL2: 3x3 filters at stride 2
[13x13x256] NORM2: Normalization layer
[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1
[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1
[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1
[6x6x256] MAX POOL3: 3x3 filters at stride 2
[4096] FC6: 4096 neurons
[4096] FC7: 4096 neurons
[1000] FC8: 1000 neurons (class scores)

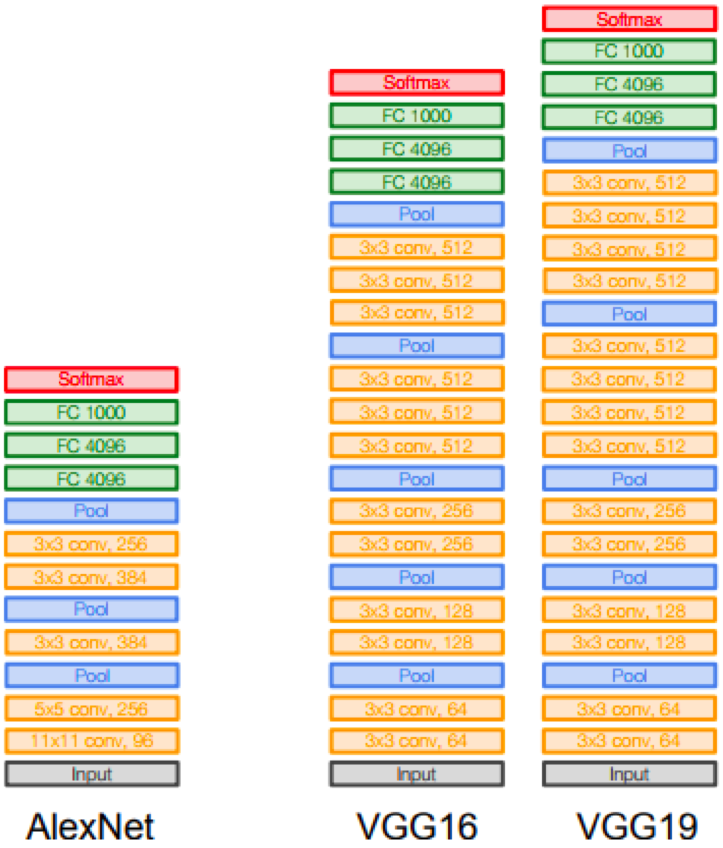
Stanford CS231

VGG, Zisserman, 2014



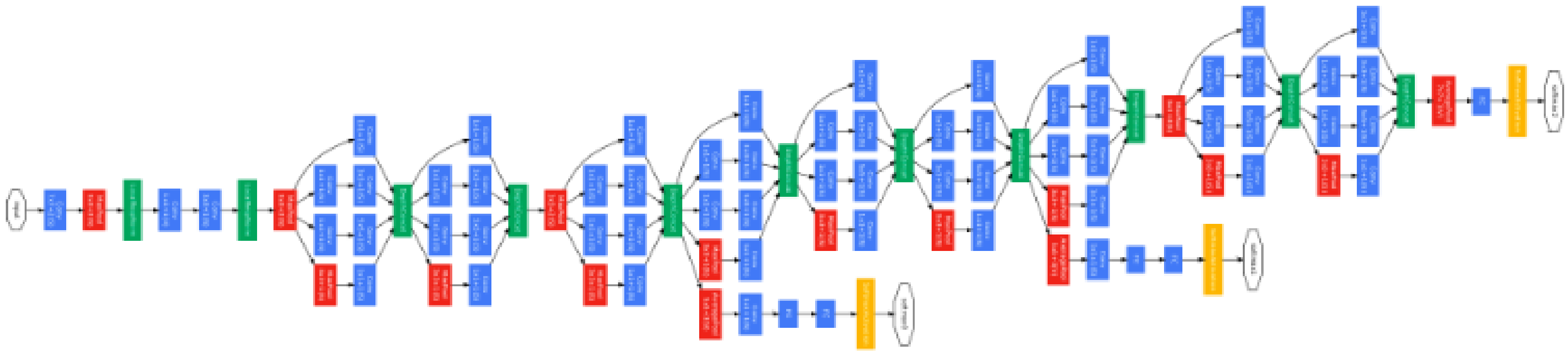
Davi Frossard

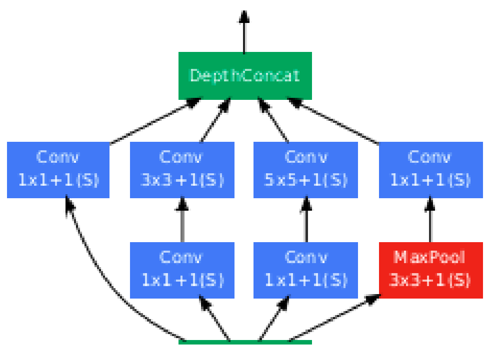
VGG



Stanford CS231

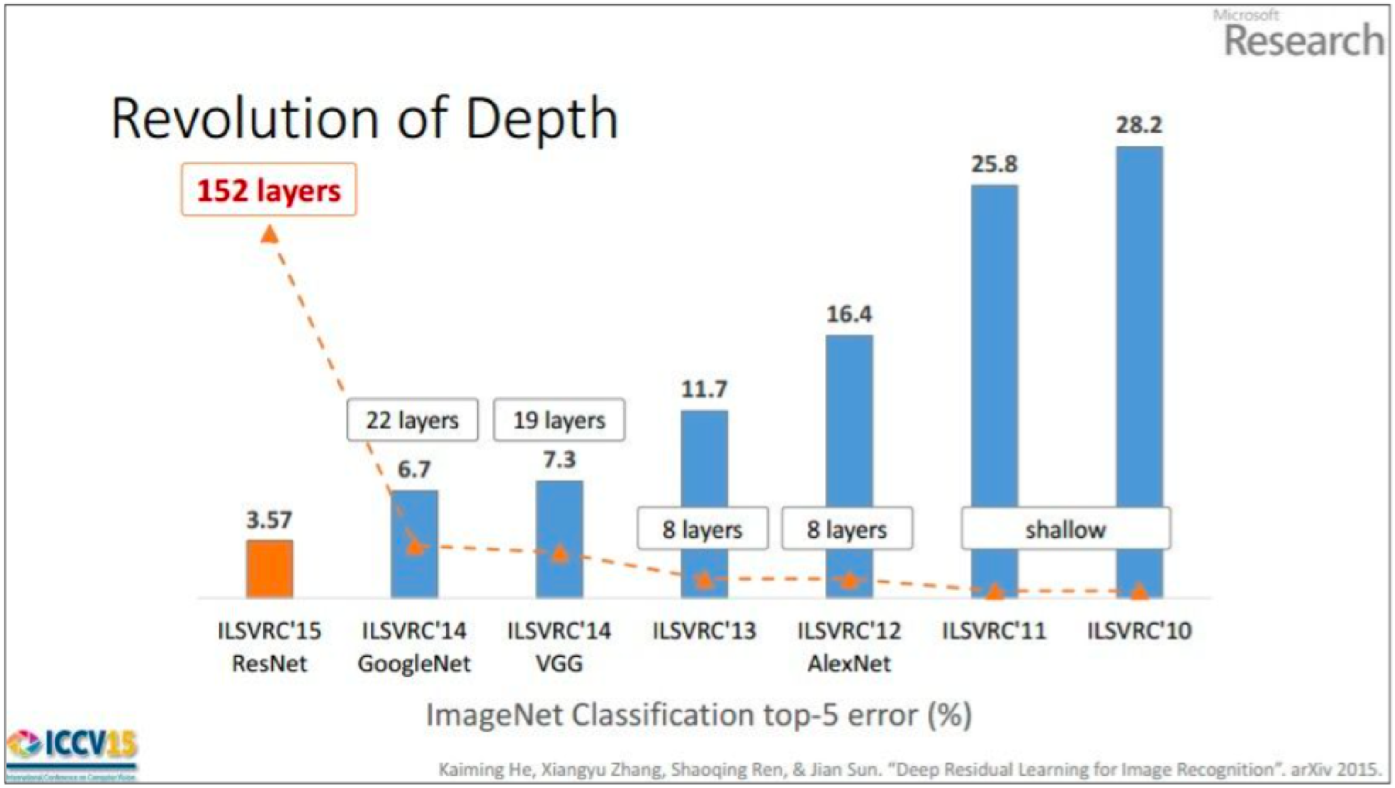
Inception, Google, 2014





Imagenet Classification

1000 kinds of objects.



(slide from Kaiming He's recent presentation)

2016 error rate is 3.0%

2017 error rate is 2.25%

Appendix: Image to Column (Im2C)

Reduce convolution to matrix multiplication — more space but faster.

$$x'[b,i,j,\Delta i,\Delta j,c_x] = x[b,i+\Delta i,j+\Delta j,c_x]$$

$$\begin{aligned}
&y[b,i,j,c_y] \\
&= \left(\sum_{\Delta i,\Delta j,c_x} W[\Delta i,\Delta j,c_x,c_y] * x[b,i+\Delta i,\ j+\Delta j,\ c_x]\right) + B[c_y] \\
&= \left(\sum_{\Delta i,\Delta j,c_x} x'[b,i,j,\Delta i,\Delta j,c_x] * W[\Delta i,\Delta j,c_x,c_y]\right) + B[c_y] \\
&= \left(\sum_{(\Delta i,\Delta j,c_x)} x'[(b,i,j),(\Delta i,\Delta j,c_x)] * W[(\Delta i,\Delta j,c_x),c_y]\right) + B[c_y]
\end{aligned}$$

Appendix: Dilation

We can "dilate" the filter by introducing an image step size d for each step in the filter coordinates.

$$y[b,i,j,c_y] \;=\; W[\Delta i, \Delta j, c_x, c_y]x[b, i + d * \Delta i, j + d * \Delta j, c_x] + B[c_y]$$

END