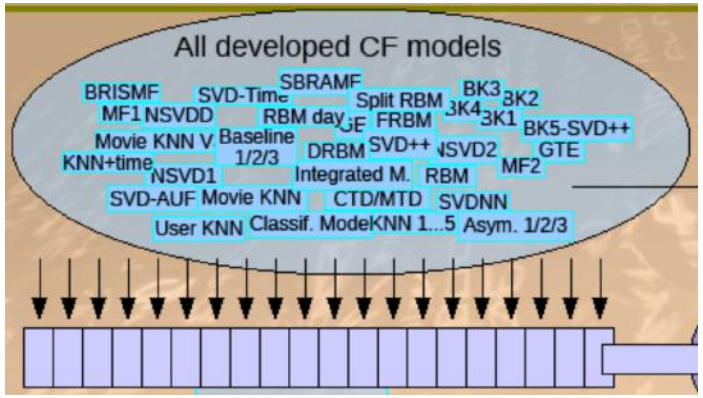
LeNet to ResNet: A Deep Journey

Journey of Improving Accuracy

Competition	Accuracy Measure	Start	Finish	Improvement	Duration
Netflix (2006 – 2009)	MSE	0.95	0.85	10.05%	3 years
ImageNet (2010 – 2015)	Error Rate	28.2	3.57	**87.34%**	5 years

Winning Strategy?



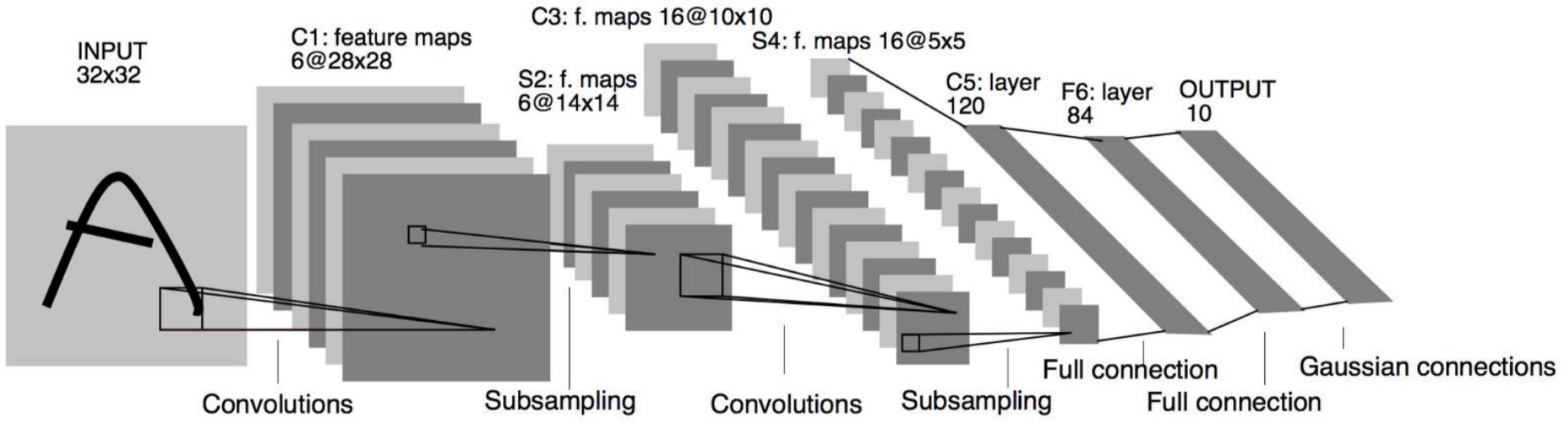
Netflix never used its $1Million algorithm

VS.

Deeper and Simpler

Deep Learning is a household name

LeNet5 (1998): The origin of convolutional neural network



Characteristics	Key Contributions	The Gap
• Repeat of Convolution – Pooling – Non Linearity • Average pooling • Sigmoid activation for the intermediate layer • tanh activation at F6 • 5x5 Convolution filter • 7 layers and less than 1M parameters	• Use of convolution to extract spatial features • Subsample using spatial average of maps • Sparse connection matrix between layers to avoid large computational cost	• Slow to train • Hard to train (Neurons dies quickly) • Lack of data

ImageNet Classification

- **ImageNet** is an image database organized according to the WordNet hierarchy
  - is formally a project aimed at (manually) labeling and categorizing images

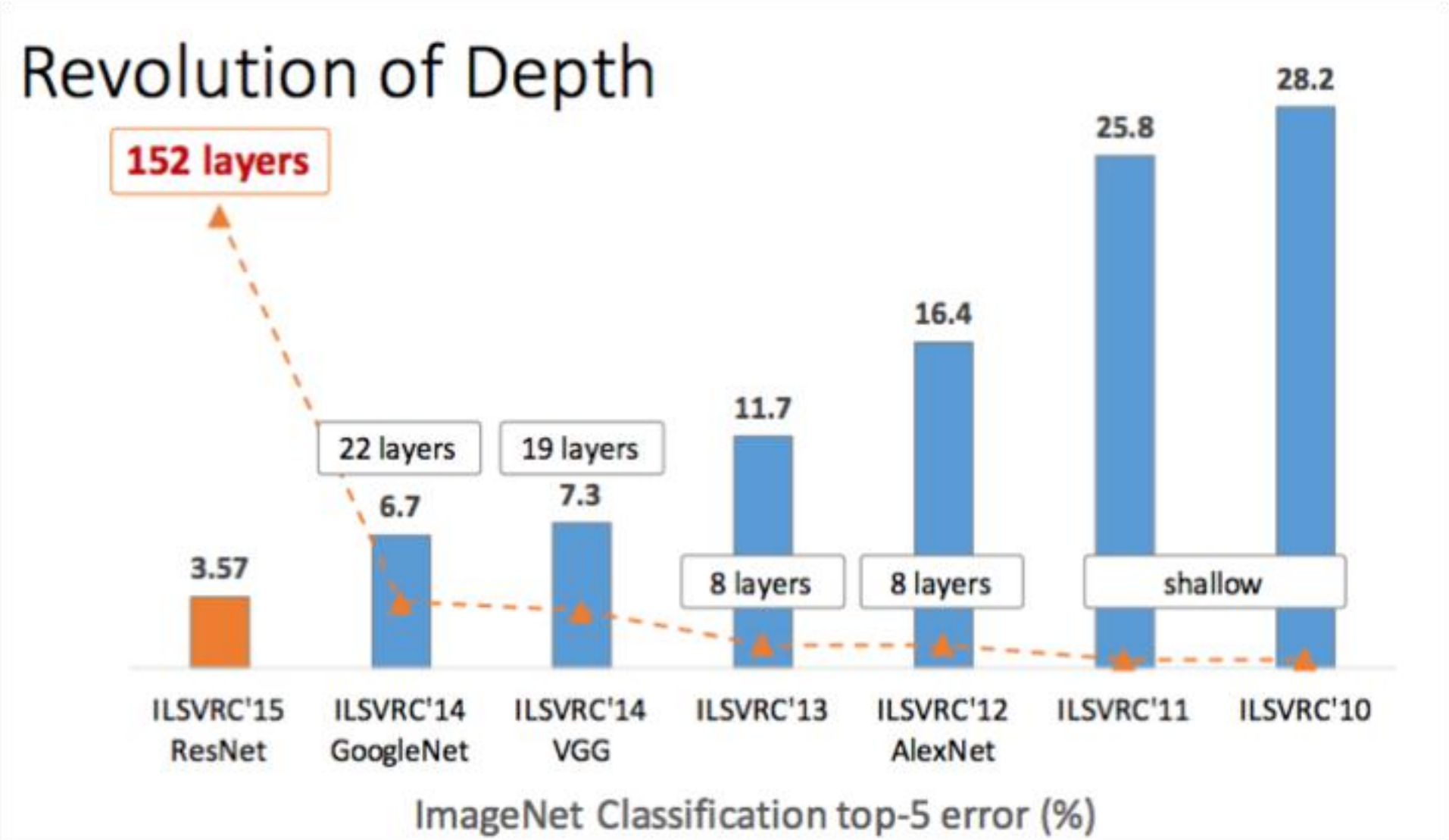
- ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

- Training Data: 1.2 Million Images, 1000+ categories
- Validation and Test Data: 150K Images, 50K Validation, Remaining Test

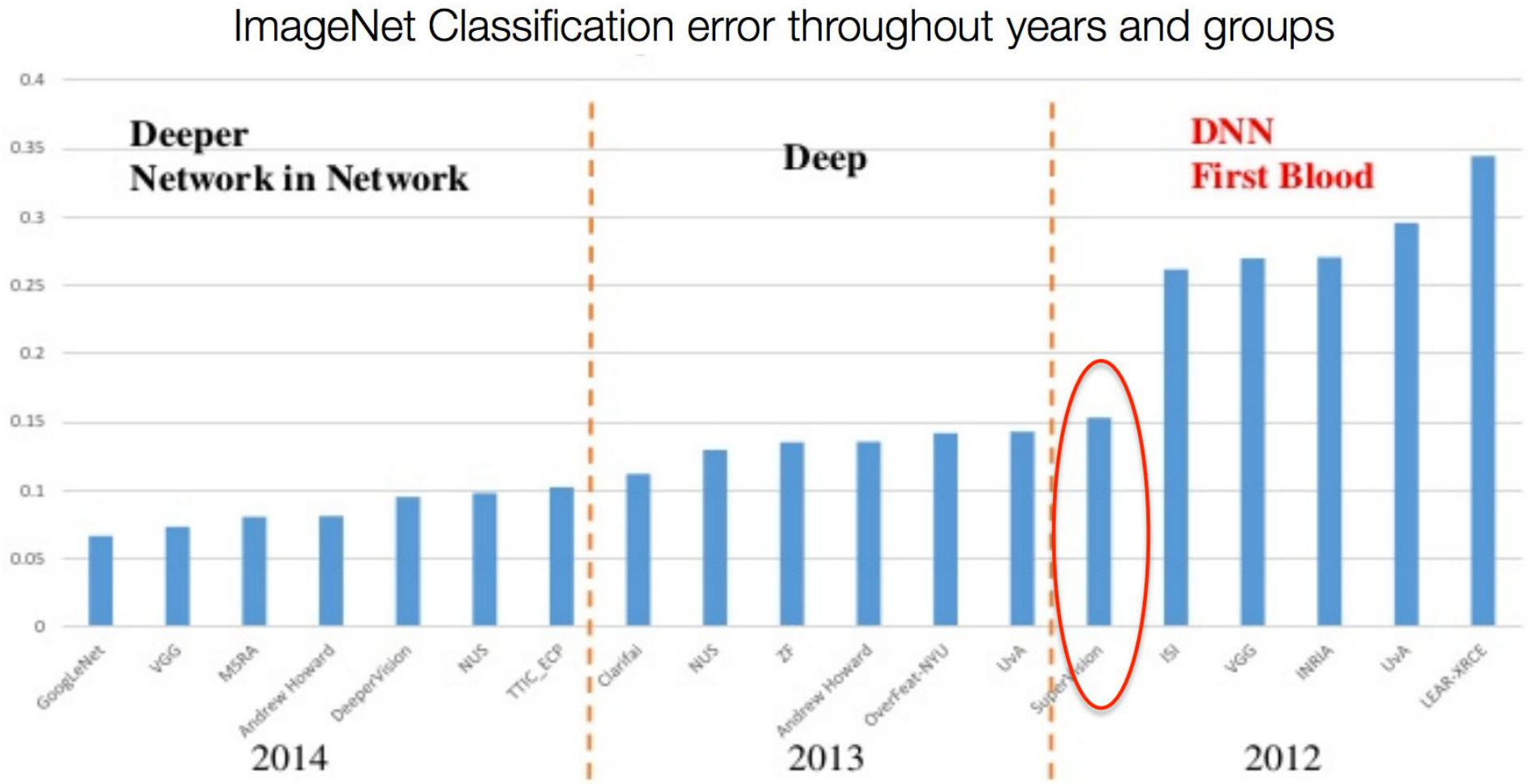
- Image Net Data: http://image-net.org/challenges/LSVRC/2010/browse-synsets

- Multiple Challenges; Object recognition, localization etc.

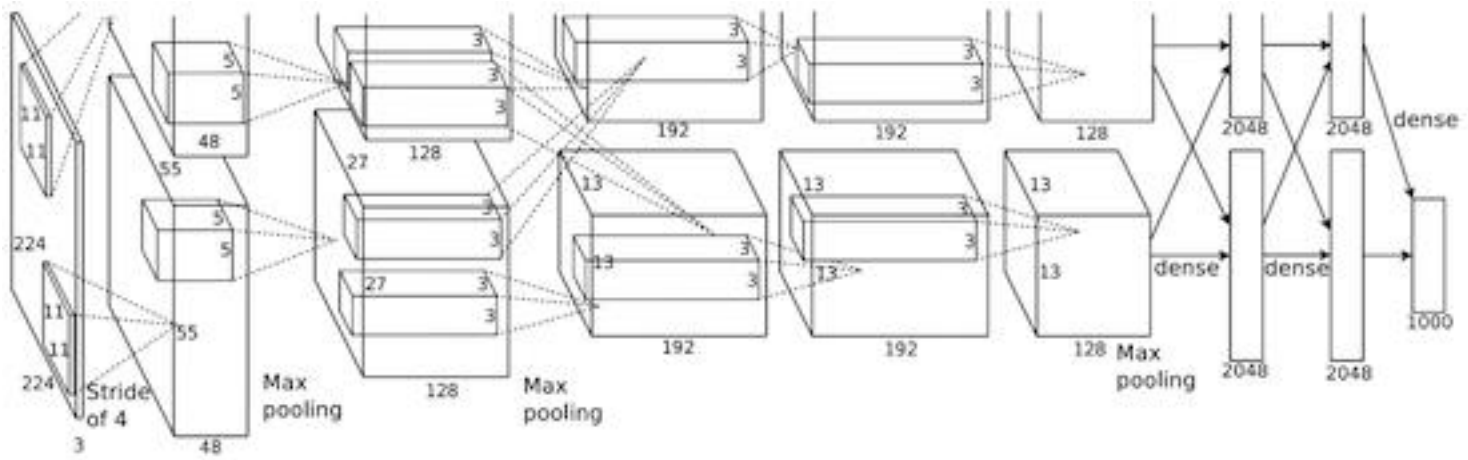
- **Fun fact**: *In 2015, Baidu scientists were banned for a year for using different accounts to greatly exceed the specified limit of two submissions per week.*


Revolution of Depth
152 layers
22 layers
19 layers
8 layers
8 layers
shallow
3.57
6.7
7.3
11.7
16.4
25.8
28.2
ILSVRC'15 ResNet
ILSVRC'14 GoogleNet
ILSVRC'14 VGG
ILSVRC'13
ILSVRC'12 AlexNet
ILSVRC'11
ILSVRC'10
ImageNet Classification top-5 error (%)

2012: The beginning of Deep

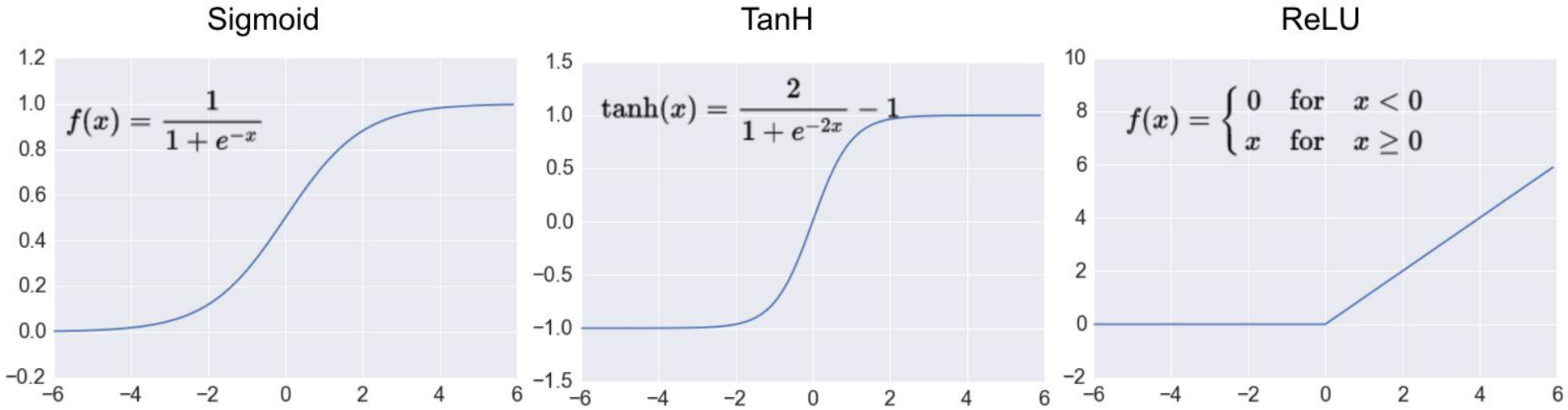


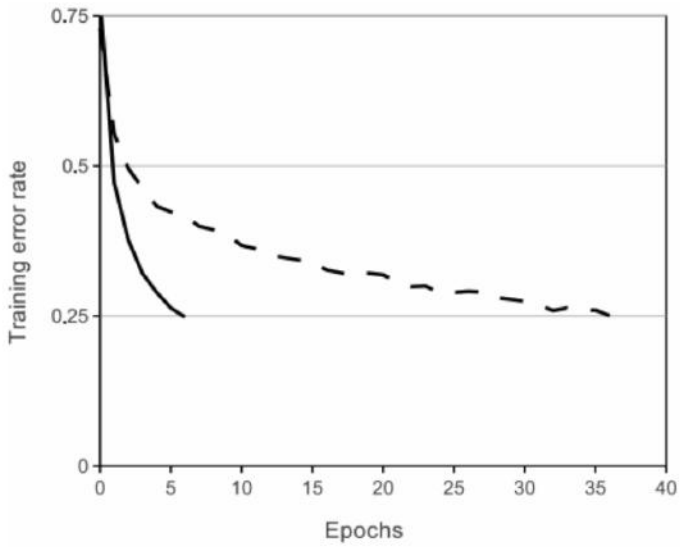
AlexNet (2012)



Characteristics	Key Contributions
- 11x11, 5x5 and 3x3 Convolutions - Max pooling - 3 FC layers - 60 Million parameters	• GPU and training in parallel • ReLu Activation • Dropout regularization • Image Augmentation

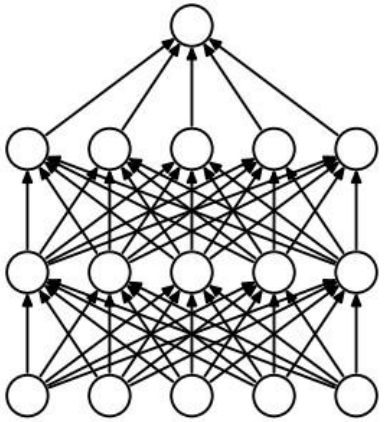
ReLU Non-Linearity – Simpler Activation



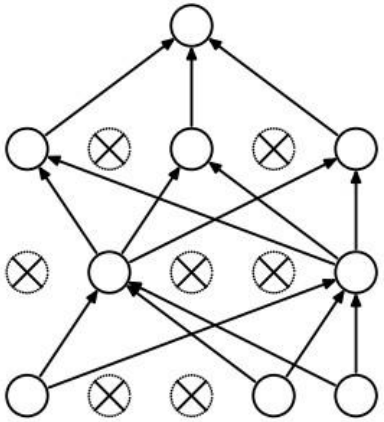


A 4 layer CNN with ReLUs is **6 times** faster than equivalent network with thanh in reaching 25% error rate on CIFR-10 dataset

Dropout – Simpler Regularization



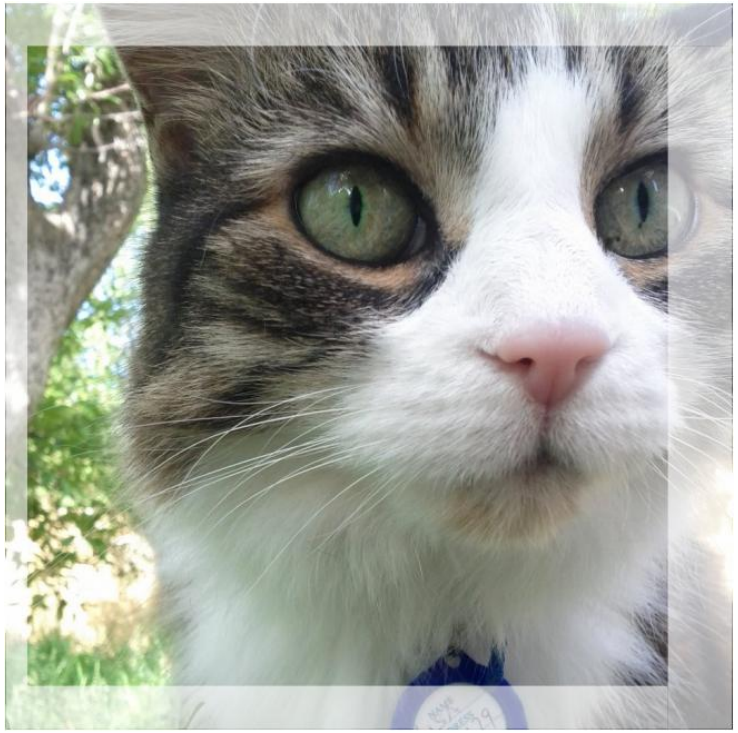
(a) Standard Neural Net



(b) After applying dropout.

```
""" Vanilla Dropout: Not recommended implementation (see notes below) """

p = 0.5 # probability of keeping a unit active. higher = less dropout

def train_step(X):
  """ X contains the data """

  # forward pass for example 3-layer neural network
  H1 = np.maximum(0, np.dot(W1, X) + b1)
  U1 = np.random.rand(*H1.shape) < p # first dropout mask
  H1 *= U1 # drop!
  H2 = np.maximum(0, np.dot(W2, H1) + b2)
  U2 = np.random.rand(*H2.shape) < p # second dropout mask
  H2 *= U2 # drop!
  out = np.dot(W3, H2) + b3

  # backward pass: compute gradients... (not shown)
  # perform parameter update... (not shown)

def predict(X):
  # ensembled forward pass
  H1 = np.maximum(0, np.dot(W1, X) + b1) * p # NOTE: scale the activations
  H2 = np.maximum(0, np.dot(W2, H1) + b2) * p # NOTE: scale the activations
  out = np.dot(W3, H2) + b3
```

Data Augmentation



- Extract 224x224 patches randomly from 256x256 images
  - Also, take their horizontal reflection
- During test time average the predictions on 5 patches and their reflection

VGG (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

- Smaller size convolution 3x3 throughout the net
- Sequence of 3x3 convolution can emulate larger receptive fields, e.g., 5x5 or 7x7
- Use of 1x1 convolution
  - Decrease in spatial volume and increase in depth of input

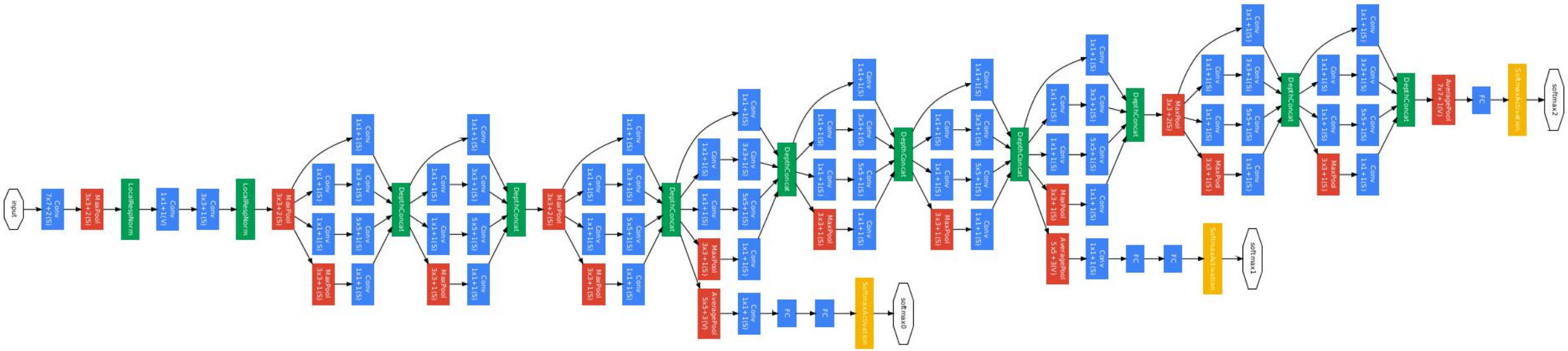
What's the advantage of using 3 layers of 3x3 instead of one layer of 7x7?

- 3 non-linear rectification layers
- Less number of parameters, $27C^2$ as opposed to $49C^2$

Key Points

- Depth is important
- Simplify the network to go deep
- 140M parameters (mostly due to the FC layers)

GoogleNet or Inception (2014)

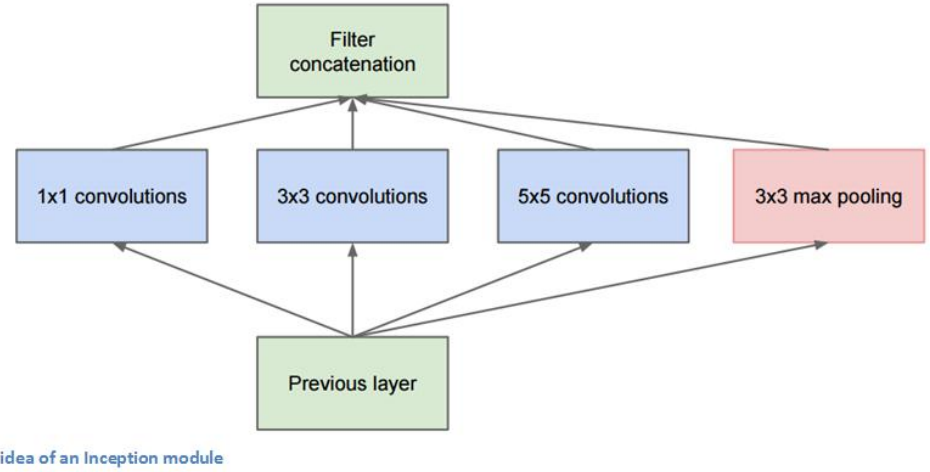


- 22 Layer CNN
- Heavy use of 1x1 ‘Network in Network’
- Use of average pooling before the classification
- Auxiliary classifiers connected to intermediate layers
  - During training add the loss of the auxiliary classifiers with a discount (0.3) weight

GoogleNet Key Ideas

- At each layer of traditional ConvNet you have to decide whether to do pooling or convolution operation, if convolution then filter size.
- **Perform all these operations in parallel**

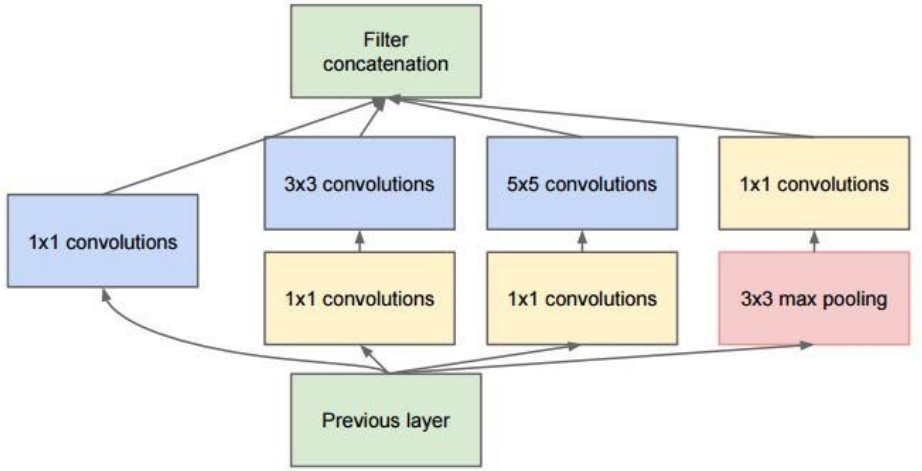
Way too many output!!!



Naïve idea of an Inception module

Naïve Version

Use 1x1 for dimensionality reduction



Modified Idea

Why 1x1 convolution?

- Introduced as “Network in Network” in 2014
- Is a way to increase Non-Linearity and spatially combine features across feature maps

Only 4M parameters compared to 60M in AlexNet

ResNet (Residual Neural Network) (2015)

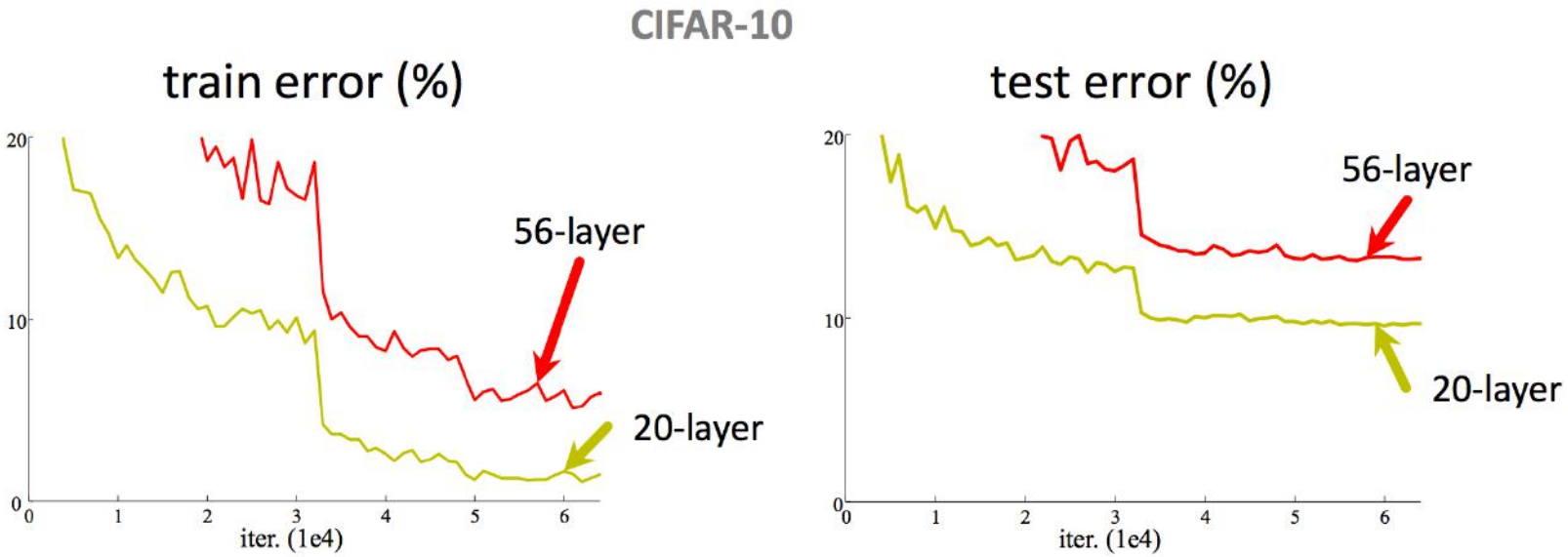
Well said Leo, well said

Why go deeper?

- Wide network is good for memorization, but not so good for generalization
- Multiple layers can learn features at various levels of abstraction
- Deep layers can provide features with global semantic meaning and abstract details (relations of relations ... of relations of objects), while using only small kernels
- Small kernels keep the number of parameters less

How to go deep?

- Simply stack layers after layers



- Plain nets: stacking of 3x3 conv layers
- 56-layers got higher training and test error

Problem of going deeper?

- Vanishing Gradient
  - Most neurons will die during back propagating the weights.

Residual Learning

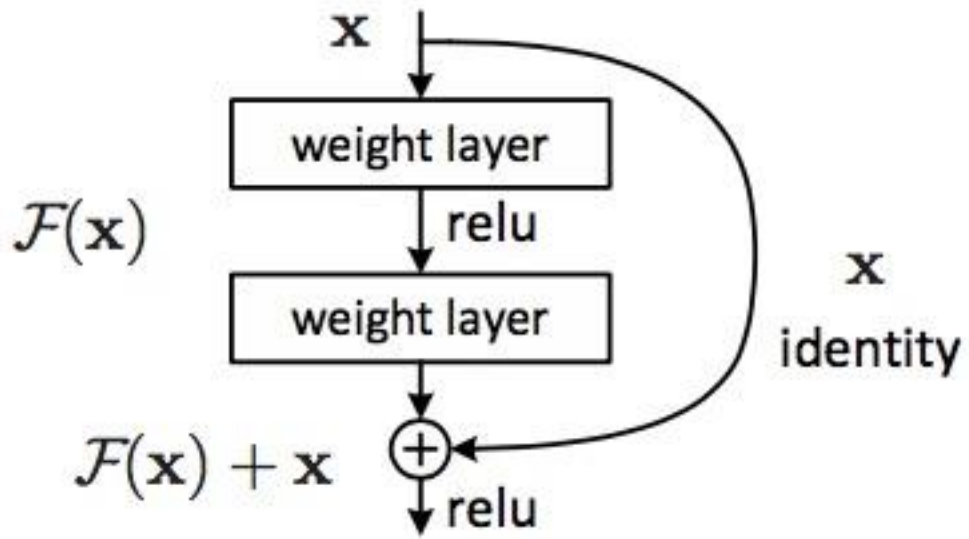
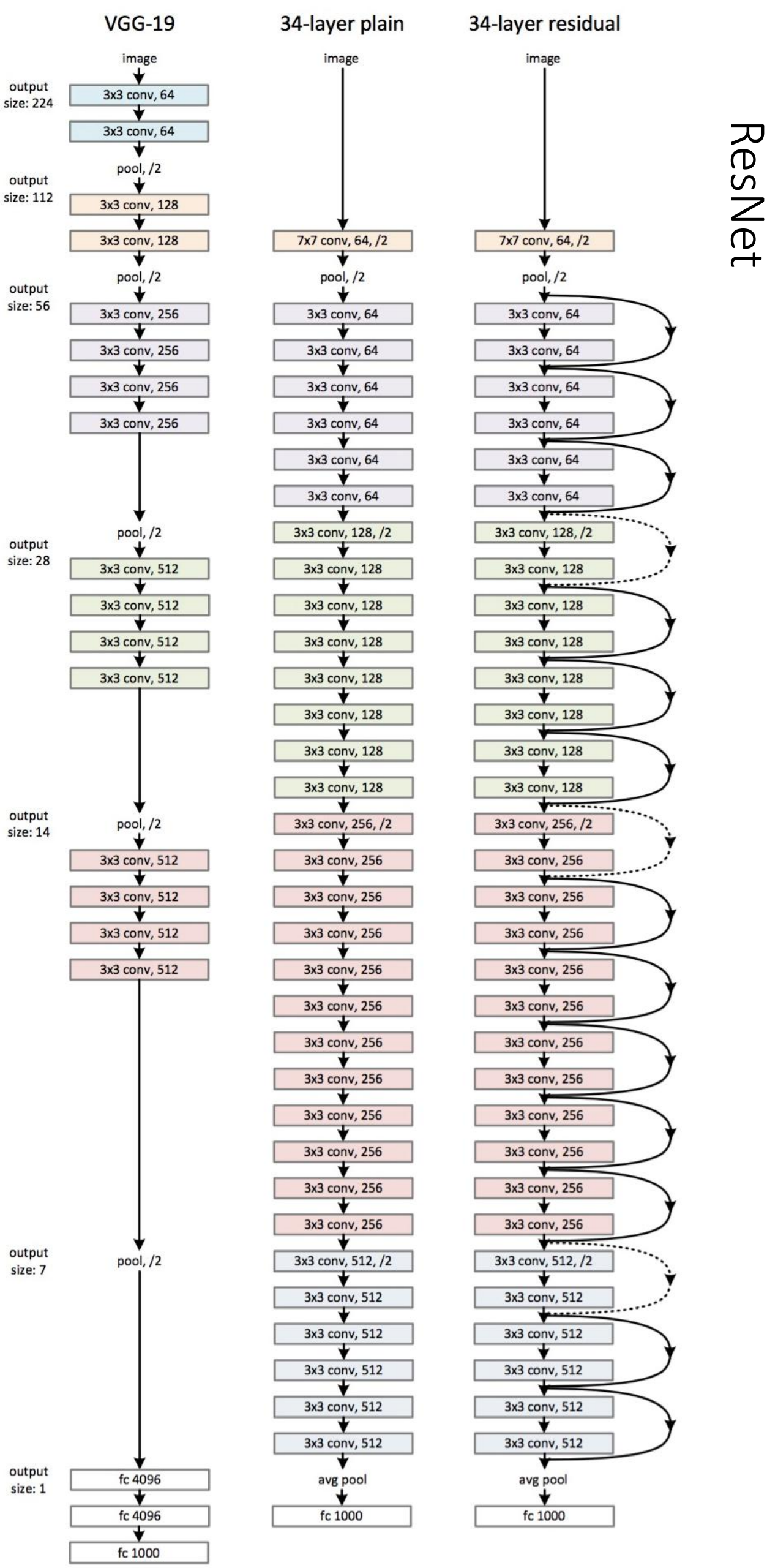
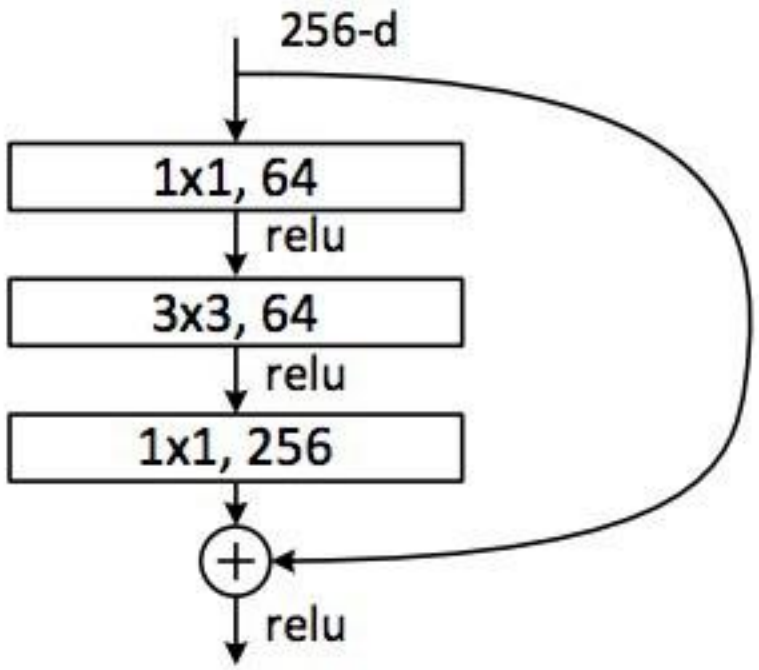


Figure 2. Residual learning: a building block.

- Introduce shortcut connections (exists in prior literature in various forms)
- Key invention is to skip 2 layers. Skipping single layer didn't give much improvement for some reason

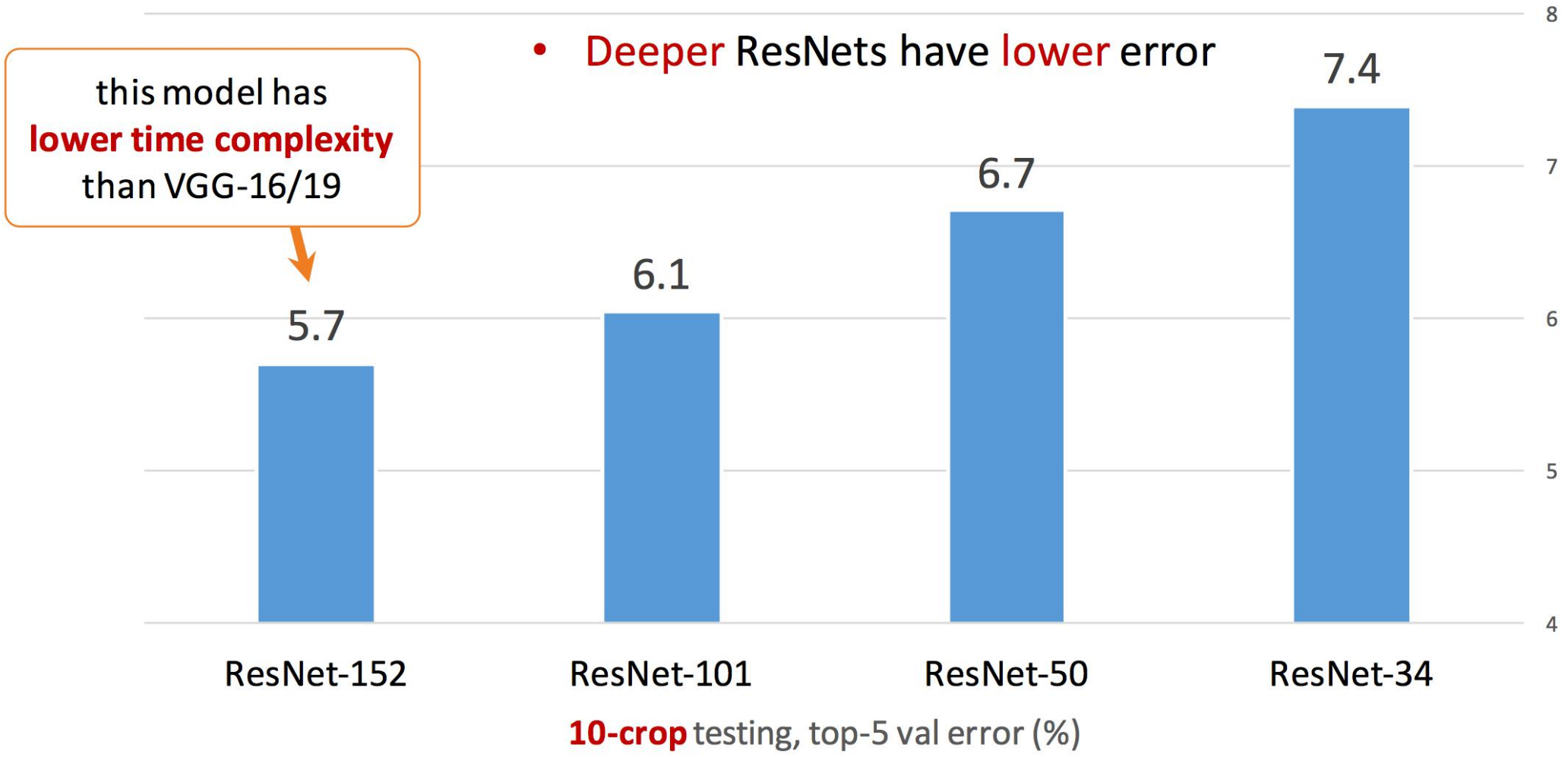

VGG-19
34-layer plain
34-layer residual
ResNet
image
output size: 224
3x3 conv, 64
3x3 conv, 64
pool, /2
output size: 112
3x3 conv, 128
3x3 conv, 128
pool, /2
output size: 56
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
pool, /2
output size: 28
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
pool, /2
output size: 14
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
pool, /2
output size: 7
fc 4096
fc 4096
fc 1000
output size: 1
7x7 conv, 64, /2
pool, /2
3x3 conv, 64
3x3 conv, 128, /2
3x3 conv, 128
3x3 conv, 256, /2
3x3 conv, 256
3x3 conv, 512, /2
3x3 conv, 512
avg pool
fc 1000

BottleNeck: A practical design



- # parameters
  - 256 x 64 + 64 x 3 x 3x 64 + 64 x 256 = ~70K
- # parameters just using 3 x 3 x 256 x 256 conv layer = ~600K

Deeper the better!!!




Deep bro, too deep I'm
drowning.

Reference

- LeNet5: http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf
- VGG: https://arxiv.org/pdf/1409.1556.pdf
- ResNet Tutoral: http://kaiminghe.com/icml16tutorial/icml2016_tutorial_deep_residual_networks_kaiminghe.pdf
- http://vision.stanford.edu/teaching/cs231b_spring1415/slides/alexnet_tugce_kyunghee.pdf
- http://slazebni.cs.illinois.edu/spring17/lec01_cnn_architectures.pdf