Carnegie Mellon University

Introduction to Deep Learning for Engineers

Spring 2025, Deep Learning for Engineers
Feb 11, 2025, Nineth Session

Amir Barati Farimani
Associate Professor of Mechanical Engineering and Bio-Engineering
Carnegie Mellon University

Story so far on CNN:

1. Concept of Representation + Learning

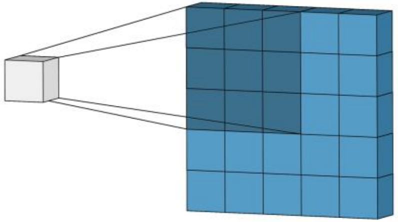
We need a UNIVERSAL REPRESENTATION LEARNER to ease the learning

2. How to learn robust representation?

We need a representation learner that can automatically learn the features needed for the task

3. How to learn spatial, high level features (Swan example)

Since high level features are spatial (not pixels) we need a scanner to scan patches of data instead of single pixels



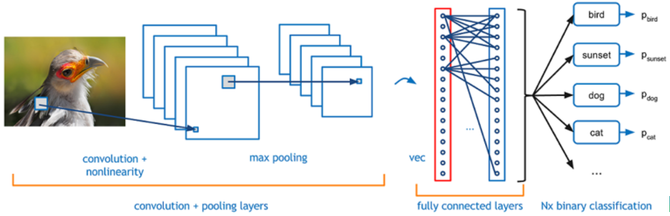
Story so far on CNN:

4. How to build a scanner for feature learning? what should be the properties of this scanner?

1. It should be numbers (a matrix) because it should be machine readable
2. It should be learnable
3. It should be flexible in size and dimension
4. Should be pluggable to Neural Networks

5. Can we design the scanners based on the learning tasks?

Yes, and we should. Because the mode of data might be different (sound, image, video) and features are needed based on the task to make a good model, the scanner should ONLY learn the relevant features connecting them to the output



Carnegie Mellon University

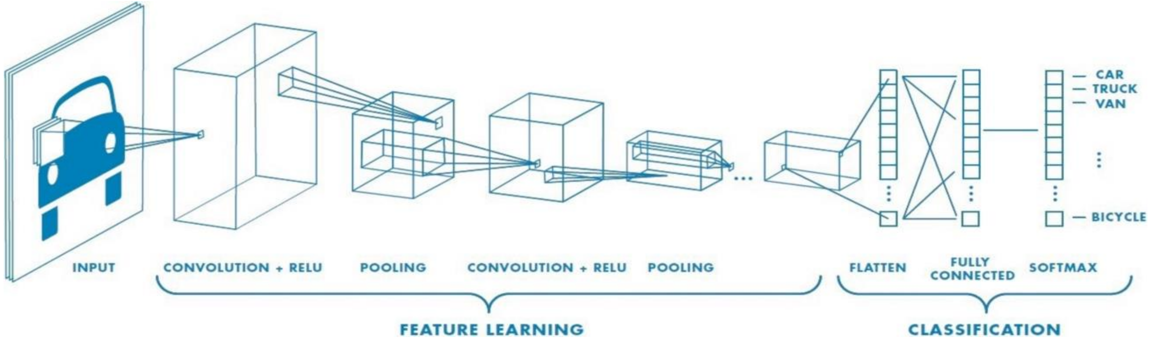
Story so far on CNN:

6. How can the scanners learn?

Inspired by iterative optimization and backpropagation in neural networks, we can iteratively learn the initialized weights of scanners (remember these are numbers)

7. How can we plug in the scanners into Neural networks?

By flattening the output of the last convolved map and passing it to the FC layer, we can forward propagate, and we can backpropagate to learn the parameters of a filter (scanner)



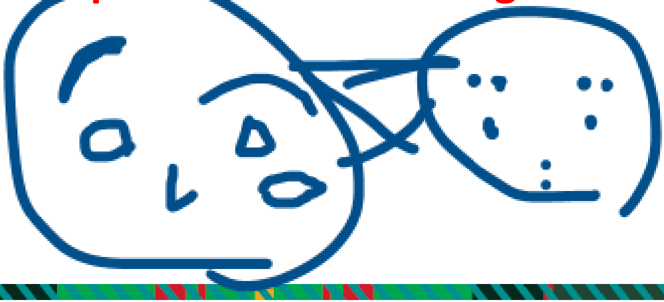
Story so far on CNN:

8. What are the components of CNN and why they are necessary?

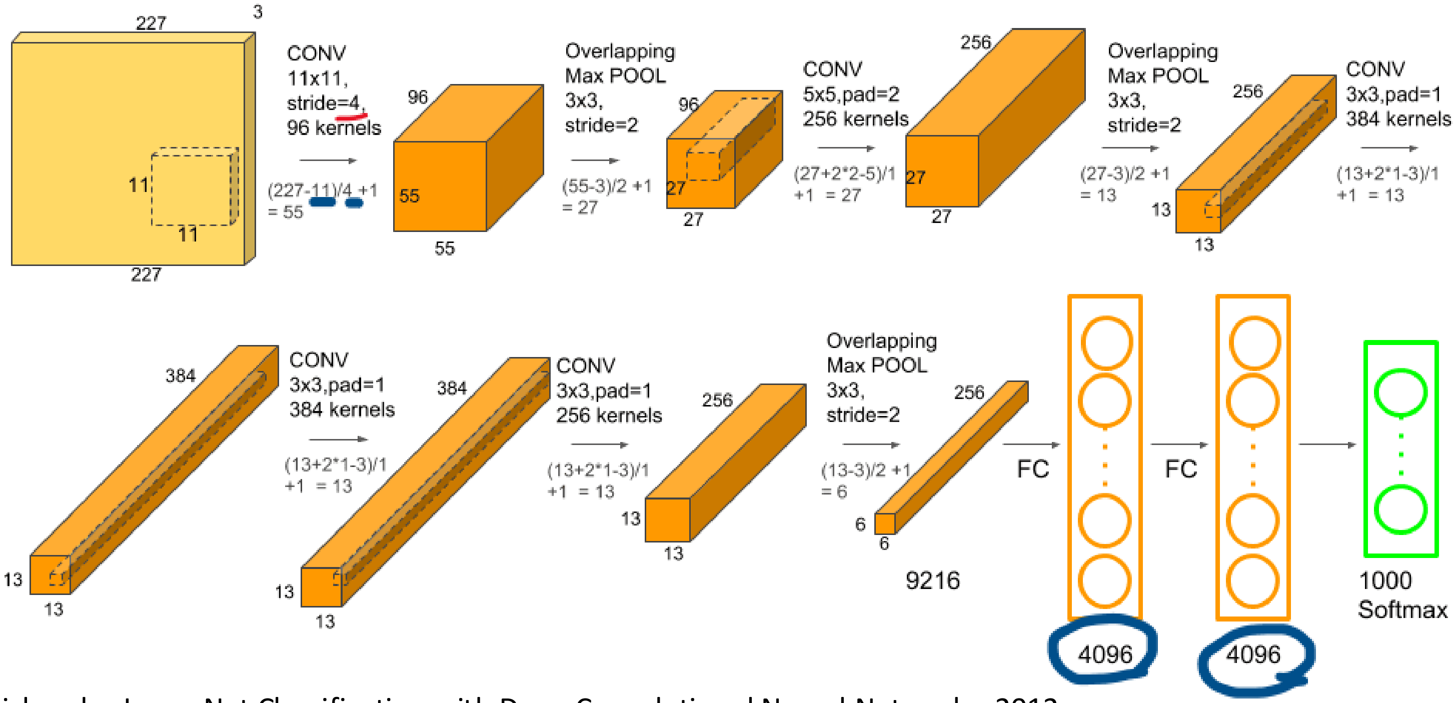
Components of CNNs are (Convolution, Non-linearity, Pooling (subsampling)). Convolution operation is for learning the filters and scanners. Non-linearity is for having more robust representation and pooling is for making the network translation invariant and focus on the important features

9. What are the good consequences of CNN layer?

1. Learning spatial features, 2. Weight sharing and reduction in the number of parameters, 3. Translation invariant representation learning



AlexNet

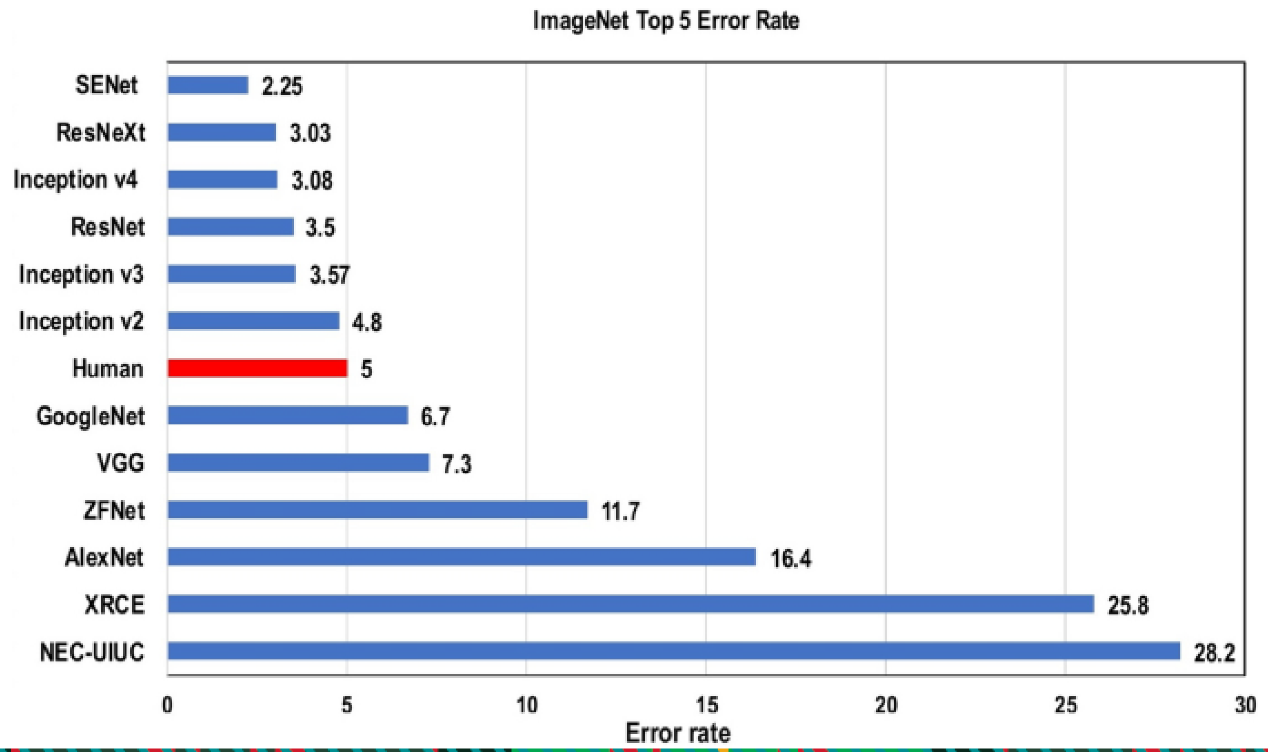


Alex Krizhevsky, ImageNet Classification with Deep Convolutional Neural Networks, 2012

Carnegie Mellon University

Different CNN Architectures

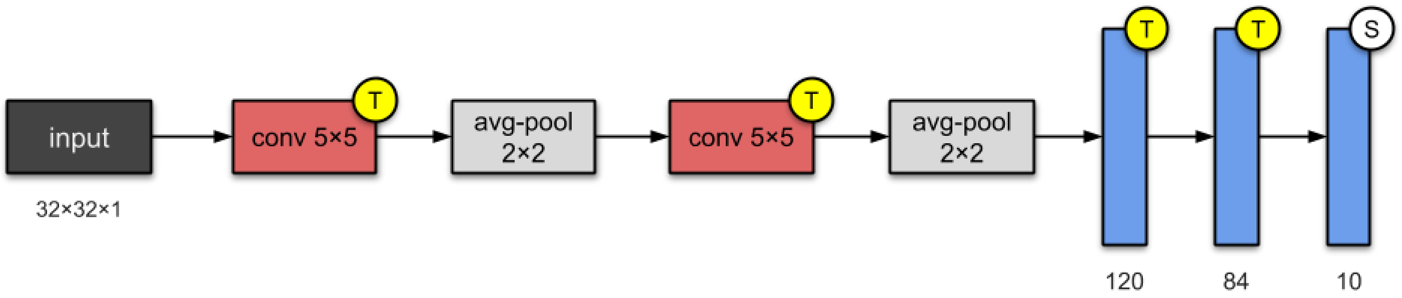
Notable CNN models[19]



Carnegie Mellon University

Notable CNN models

LENET (1998)[8]



(2) ENDING THE NETWORK WITH ONE OR MORE FULLY-CONNECTED LAYERS

LeNet-5

Gradient Based Learning Applied To Document Recognition
- Y. Lecun, L. Bottou, Y. Bengio, P. Haffner; 1998

Helped establish how we use CNNs today

Replaced manual feature

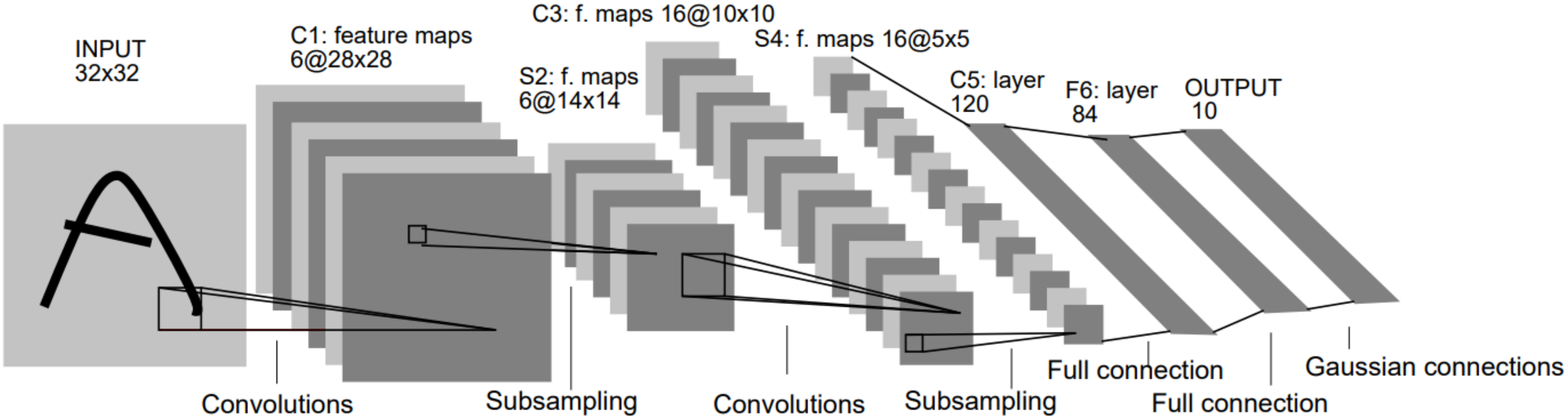
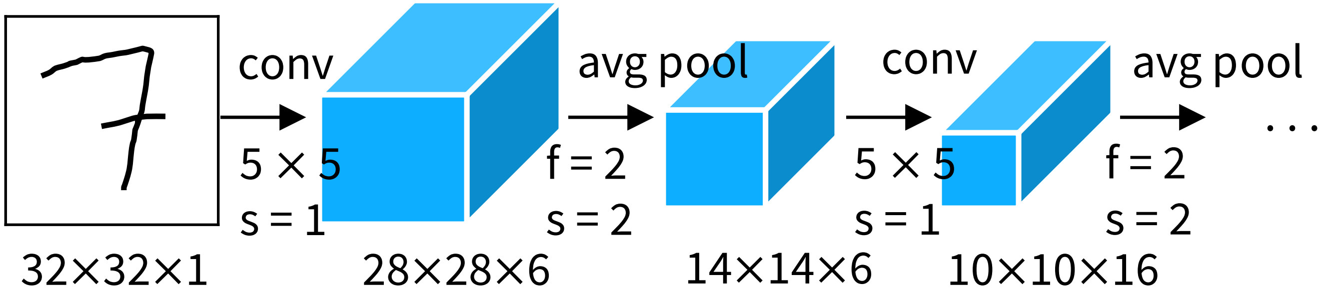


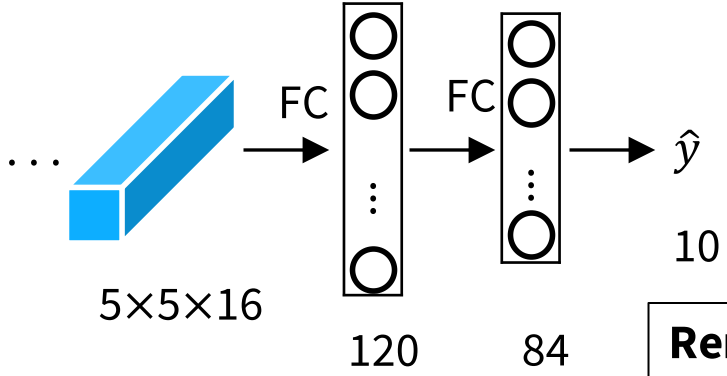
Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

Carnegie Mellon University

[**LeCun et al., 1998**]

LeNet-5





Only 60K parameters
As we go deeper in the network: $N_H \downarrow,\ N_W \downarrow,\ N_C \uparrow$
General structure:
conv->pool->conv->pool->FC->FC->output

Different filters look at different channels
Sigmoid and Tanh nonlinearity

Reminder:
Output size = (N+2P-F)/stride + 1

Carnegie Mellon University

This slide is taken from Andrew Ng

[**LeCun et al., 1998**]

AlexNet

ImageNet Classification with Deep Convolutional Neural Networks

- *Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton; 2012*

Facilitated by GPUs, highly optimized convolution implementation and large datasets (ImageNet)

One of the largest CNNs to date

Has 60 Million parameter compared to 60k parameter of LeNet-5

[**Krizhevsky et al**., **2012**]

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners

The annual "Olympics" of computer vision.

Teams from across the world compete to see who has the best computer vision model for tasks such as classification, localization, detection, and more.

2012 marked **the first year where a CNN was used** to achieve a top 5 test error rate of 15.3%.

The next best entry achieved an error of 26.2%.

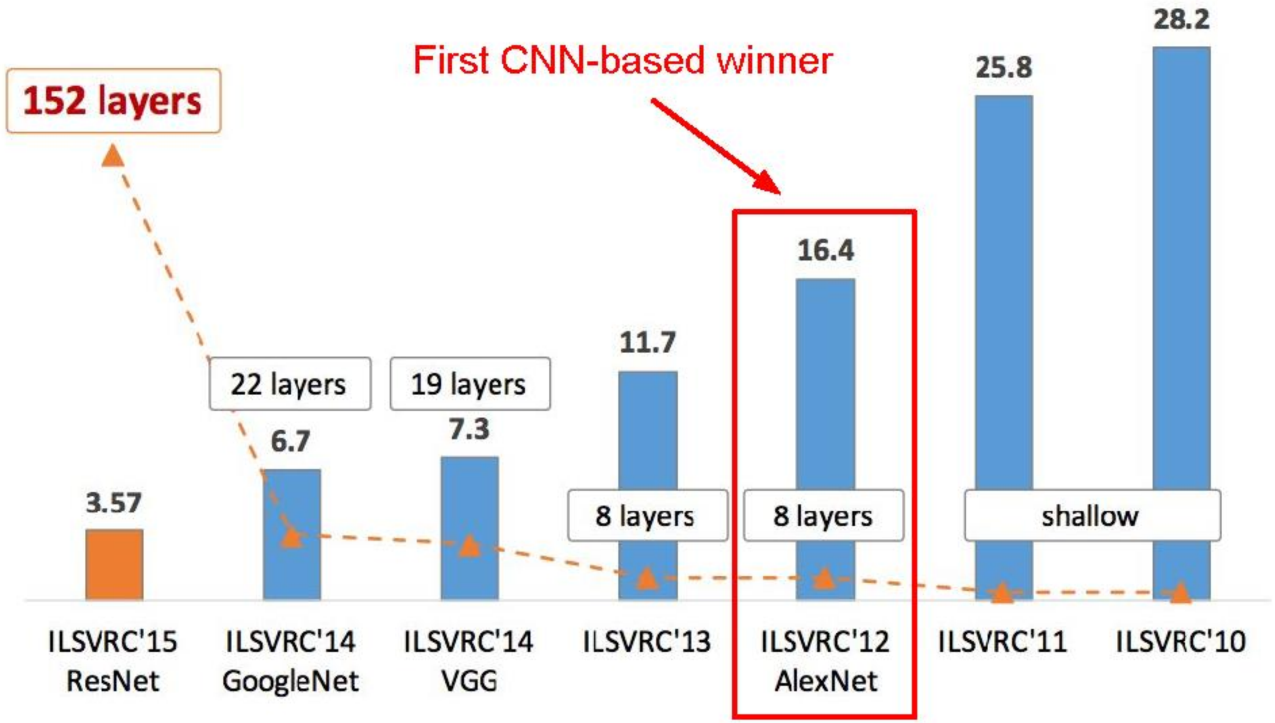
AlexNet

Model	Top-1	Top-5
Sparse coding [2]	*47.1%*	*28.2%*
SIFT + FVs [24]	*45.7%*	*25.7%*
CNN	**37.5%**	**17.0%**

Alex Krizhevsky, ImageNet Classification with Deep Convolutional Neural Networks, 2012

Carnegie Mellon University

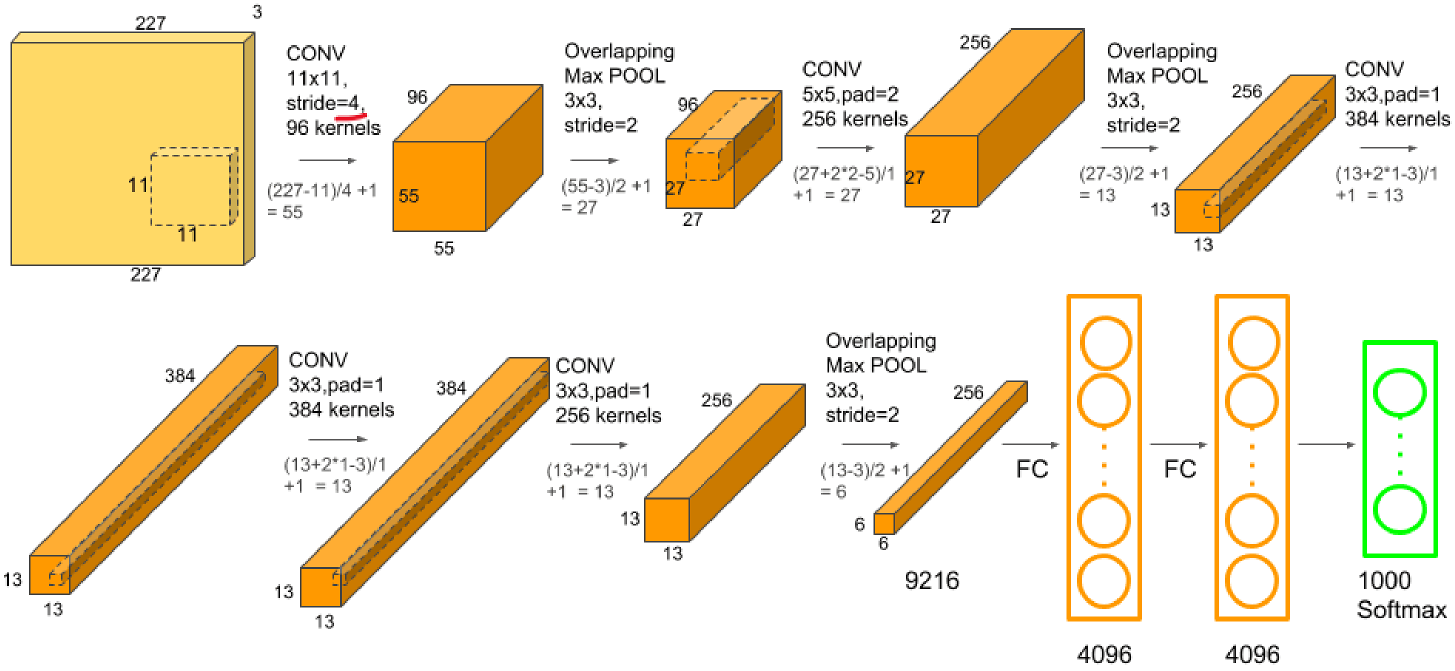
ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners



Carnegie Mellon University

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

AlexNet



Alex Krizhevsky, ImageNet Classification with Deep Convolutional Neural Networks, 2012

Carnegie Mellon University

AlexNet

Architectur
eCONV1
MAX POOL1
NORM1
CONV2
MAX POOL2
NORM2
CONV3
CONV4
CONV5
Max POOL3
FC6
FC7
FC8

INPUT: 227X227X3 IMAGES (224X224 BEFORE PADDING)

FIRST LAYER: 96 11X11 FILTERS APPLIED AT STRIDE 4

OUTPUT VOLUME SIZE?

(N-F)/s+1 = (227-11)/4+1 = 55 -> [55x55x96]

NUMBER OF PARAMETERS IN THIS LAYER?

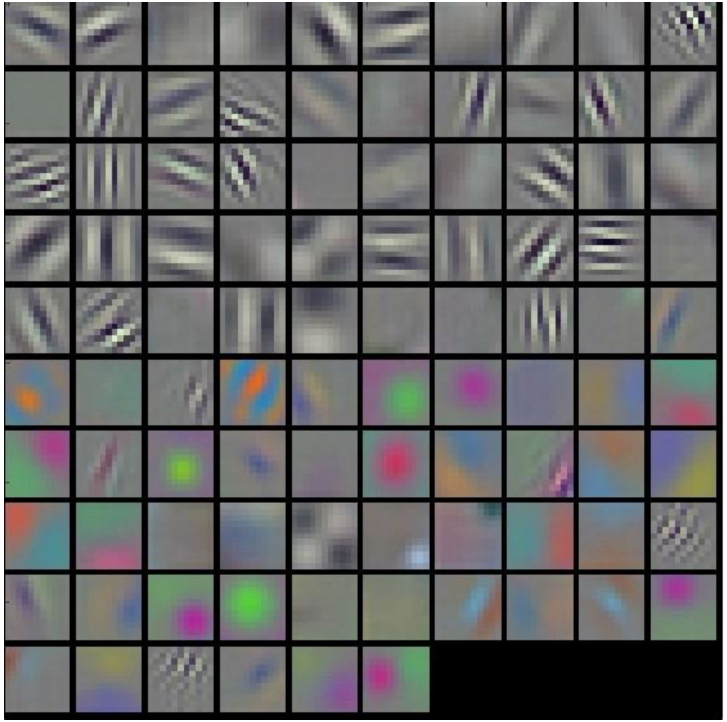
(11*11*3)*96 = 35K

Carnegie Mellon University

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[**Krizhevsky et al., 2012**]

AlexNet filters



[Krizhevsky et al., 2012]

AlexNet

Architecture
CONV1
MAX POOL1
NORM1
CONV2
MAX POOL2
NORM2
CONV3
CONV4
CONV5
Max POOL3
FC6
FC7
FC8

INPUT: 227X227X3 IMAGES (224X224 BEFORE PADDING)

AFTER CONV1: 55X55X96

SECOND LAYER: 3X3 FILTERS APPLIED AT STRIDE 2

OUTPUT VOLUME SIZE?

(N-F)/s+1 = (55-3)/2+1 = 27 -> [27x27x96]

NUMBER OF PARAMETERS IN THIS LAYER?

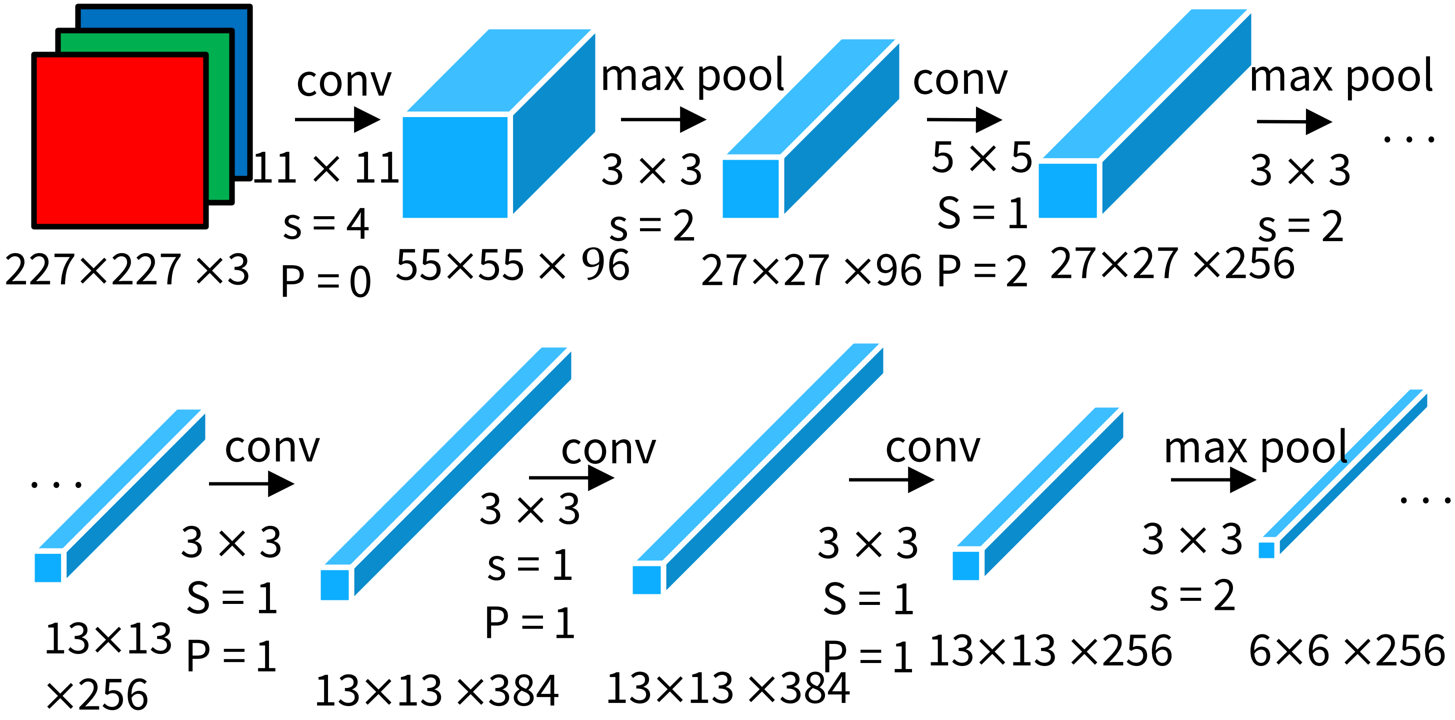
0!

Carnegie Mellon University

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[Krizhevsky et al., 2012]

AlexNet

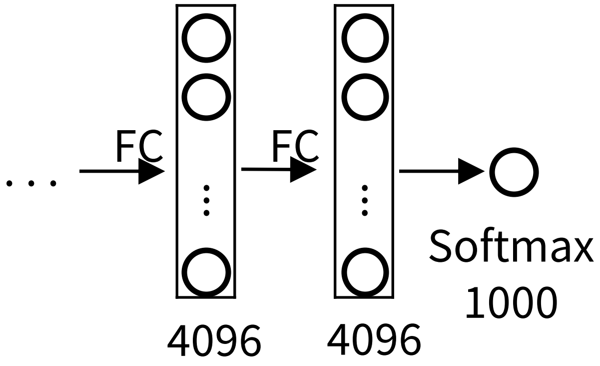


Carnegie Mellon University

This slide is taken from Andrew Ng

[Krizhevsky et al., 2012]

AlexNet



This slide is taken from Andrew Ng

[Krizhevsky et al., 2012]

DETAILS/RETROSPECTIVES:

FIRST USE OF RELU

USED NORM LAYERS (NOT COMMON ANYMORE)

HEAVY DATA AUGMENTATION

DROPOUT 0.5

BATCH SIZE 128

7 CNN ENSEMBLE

[Krizhevsky et al., 2012]

AlexNet

TRAINED ON GTX 580 GPU WITH ONLY 3 GB OF MEMORY.

NETWORK SPREAD ACROSS 2 GPUS, HALF THE NEURONS (FEATURE MAPS) ON EACH GPU.

CONV1, CONV2, CONV4, CONV5:
CONNECTIONS ONLY WITH FEATURE MAPS ON SAME GPU.

CONV3, FC6, FC7, FC8:
CONNECTIONS WITH ALL FEATURE MAPS IN PRECEDING LAYER, COMMUNICATION ACROSS GPUS.

Carnegie Mellon University

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[**Krizhevsky et al., 2012**]

AlexNet

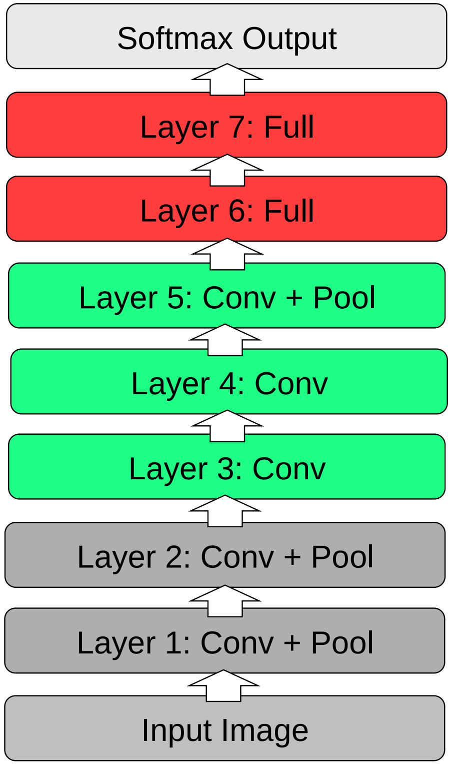
AlexNet was the coming out party for CNNs in the computer vision community. This was **the first time a model performed so well on a historically difficult ImageNet dataset.** This paper illustrated the benefits of CNNs and backed them up with record breaking performance in the competition.

Carnegie Mellon University

[**Krizhevsky et al.**, **2012**]

AlexNet

- 8 layers total!
- Trained on Imagenet dataset [Deng et al. CVPR'09]!
- 18.2% top-5 error !



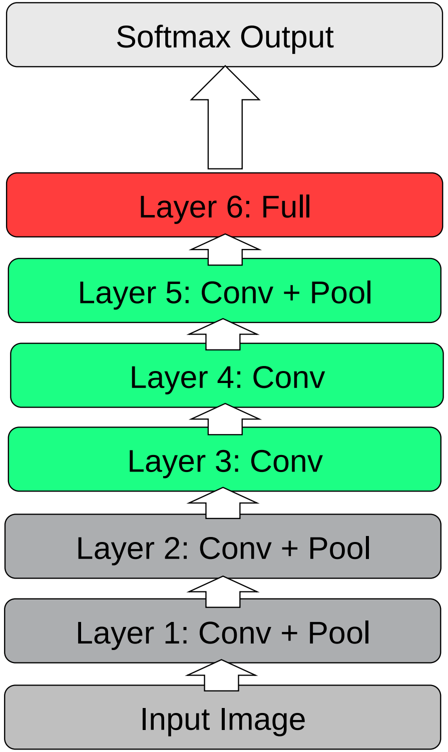
[From Rob Fergus' CIFAR 2016 tutorial] Krizhevsky et al., NIPS 2012

Carnegie Mellon University

AlexNet

AlexNet

- Remove top fully connected layer 7 !
- Drop ~16 million parameters!
- Only 1.1% drop in performance!!



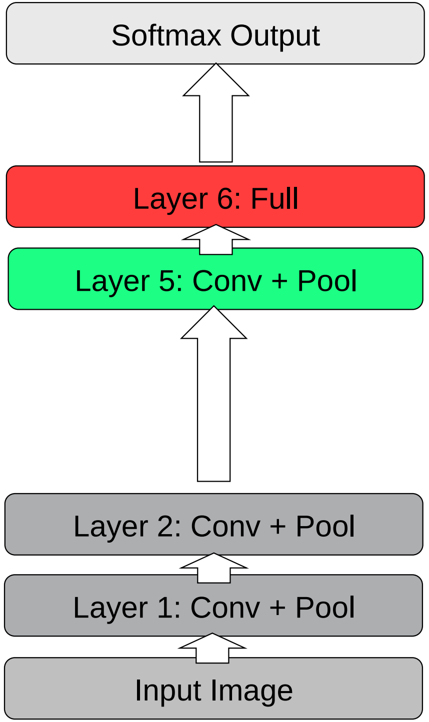
[From Rob Fergus' CIFAR 2016 tutorial]   Krizhevsky et al., NIPS 2012

Carnegie Mellon University

AlexNet

AlexNet

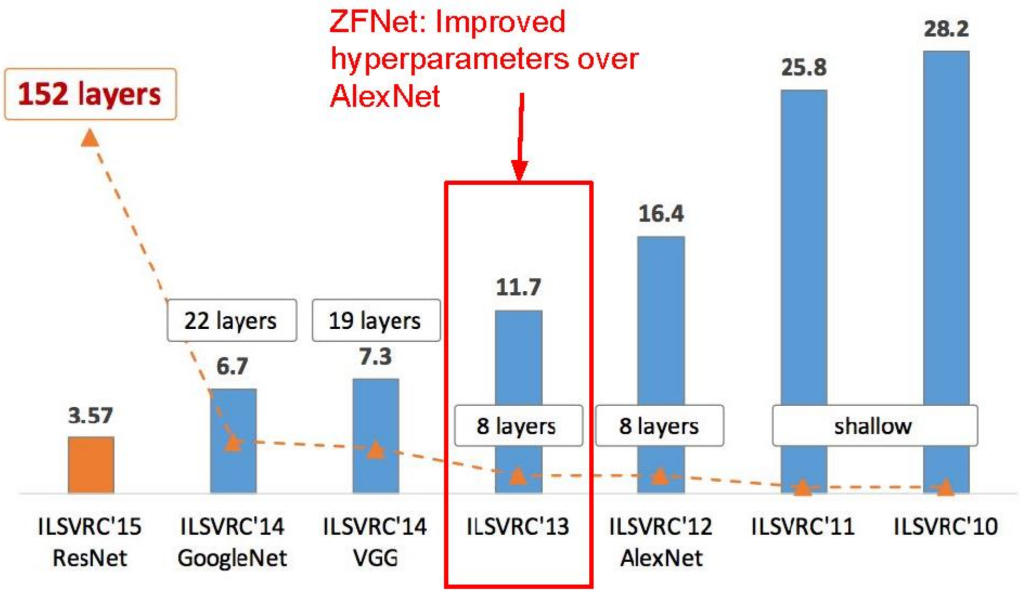
- Remove layers 3 4,6 and 7 !
- Drop ~50 million parameters!
- **33.5%** drop in performance!!
- Depth of the network is the key!



[From Rob Fergus' CIFAR 2016 tutorial] Krizhevsky et al., NIPS 2012

Carnegie Mellon University

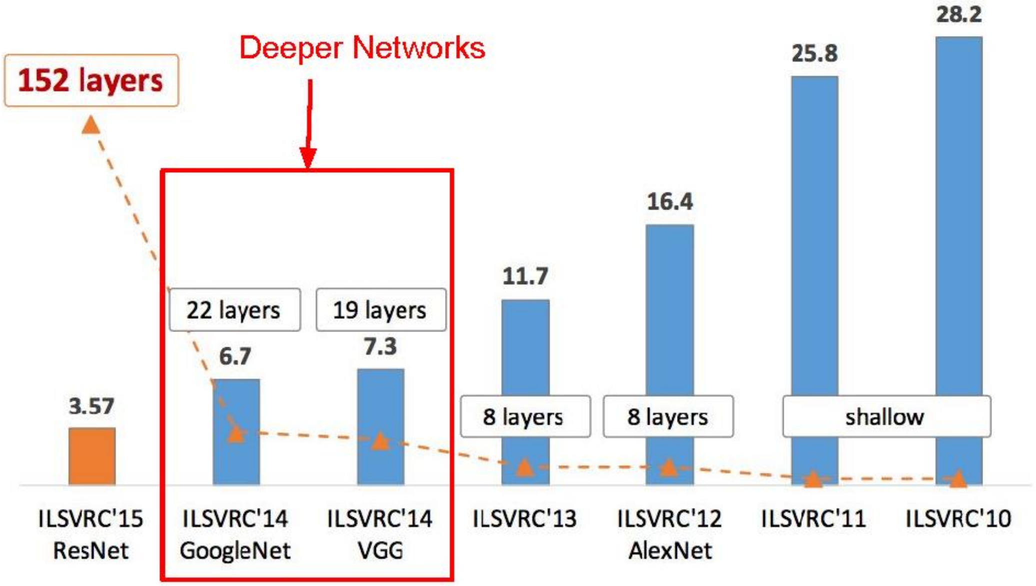
ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners



Carnegie Mellon University

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners



Carnegie Mellon University

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

VGGNet

Very Deep Convolutional Networks For Large Scale Image Recognition - Karen Simonyan and Andrew Zisserman; 2015

The runner-up at the ILSVRC 2014 competition

Significantly deeper than AlexNet

140 million parameters

[Simonyan and Zisserman, 2014]

Input
3x3 conv, 64
3x3 conv, 64
Pool 1/2
3x3 conv, 128
3x3 conv, 128
Pool 1/2
3x3 conv, 256
3x3 conv, 256
Pool 1/2
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
Pool 1/2
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
Pool 1/2
FC 4096
FC 4096
FC 1000
Softmax

VGGNet

Smaller filters

- Only 3x3 CONV filters, stride 1, pad 1 and 2x2 MAX POOL , stride 2

Deeper network

- AlexNet: 8 layers
- VGGNet: 16 - 19 layers

ZFNet: 11.7% top 5 error in ILSVRC'13

VGGNet: 7.3% top 5 error in ILSVRC'14

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[Simonyan and Zisserman, 2014]

VGGNet

WHY USE SMALLER FILTERS? (3X3 CONV)

STACK OF THREE 3X3 CONV (STRIDE 1) LAYERS HAS THE SAME EFFECTIVE RECEPTIVE FIELD AS ONE 7X7 CONV LAYER.

WHAT IS THE EFFECTIVE RECEPTIVE FIELD OF THREE 3X3 CONV (STRIDE 1) LAYERS?

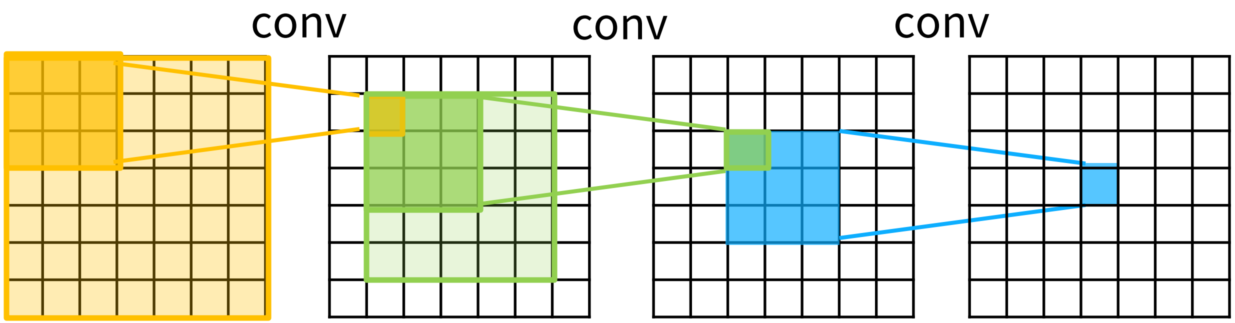
7X7

BUT DEEPER, MORE NON-LINEARITIES

AND FEWER PARAMETERS: $3 * (3^2C^2)$ VS. 7^2C^2 FOR C CHANNELS PER LAYER

[Simonyan and Zisserman, 2014]

Reminder: Receptive Field



Input	memory: 224*224*3=150K	params: 0
3x3 conv, 64	memory: 224*224*64=3.2M	params: (3*3*3)*64 = 1,728
3x3 conv, 64	memory: 224*224*64=3.2M	params: (3*3*64)*64 = 36,864
Pool	memory: 112*112*64=800K	params: 0
3x3 conv, 128	memory: 112*112*128=1.6M	params: (3*3*64)*128 = 73,728
3x3 conv, 128	memory: 112*112*128=1.6M	params: (3*3*128)*128 = 147,456
Pool	memory: 56*56*128=400K	params: 0
3x3 conv, 256	memory: 56*56*256=800K	params: (3*3*128)*256 = 294,912
3x3 conv, 256	memory: 56*56*256=800K	params: (3*3*256)*256 = 589,824
3x3 conv, 256	memory: 56*56*256=800K	params: (3*3*256)*256 = 589,824
Pool	memory: 28*28*256=200K	params: 0
3x3 conv, 512	memory: 28*28*512=400K	params: (3*3*256)*512 = 1,179,648
3x3 conv, 512	memory: 28*28*512=400K	params: (3*3*512)*512 = 2,359,296
3x3 conv, 512	memory: 28*28*512=400K	params: (3*3*512)*512 = 2,359,296
Pool	memory: 14*14*512=100K	params: 0
3x3 conv, 512	memory: 14*14*512=100K	params: (3*3*512)*512 = 2,359,296
3x3 conv, 512	memory: 14*14*512=100K	params: (3*3*512)*512 = 2,359,296
3x3 conv, 512	memory: 14*14*512=100K	params: (3*3*512)*512 = 2,359,296
Pool	memory: 7*7*512=25K	params: 0
FC 4096	memory: 4096	params: 7*7*512*4096 = 102,760,448
FC 4096	memory: 4096	params: 4096*4096 = 16,777,216
FC 1000	memory: 1000	params: 4096*1000 = 4,096,000

18M

Carnegie Mellon University

[Simonyan and Zisserman, 2014]

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

VGGNet

Input
3x3 conv, 64
3x3 conv, 64
Pool
3x3 conv, 128
3x3 conv, 128
Pool
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
Pool
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
Pool
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
Pool
FC 4096
FC 4096
FC 1000
Softmax

VGG16:

TOTAL MEMORY: 24M * 4 BYTES ~= 96MB / IMAGE

TOTAL PARAMS: 138M PARAMETERS

[Simonyan and Zisserman, 2014]

Input	memory: 224*224*3=150K	params: 0
3x3 conv, 64	memory: 224*224*64=3.2M	params: (3*3*3)*64 = 1,728
3x3 conv, 64	memory: 224*224*64=3.2M	params: (3*3*64)*64 = 36,864
Pool	memory: 112*112*64=800K	params: 0
3x3 conv, 128	memory: 112*112*128=1.6M	params: (3*3*64)*128 = 73,728
3x3 conv, 128	memory: 112*112*128=1.6M	params: (3*3*128)*128 = 147,456
Pool	memory: 56*56*128=400K	params: 0
3x3 conv, 256	memory: 56*56*256=800K	params: (3*3*128)*256 = 294,912
3x3 conv, 256	memory: 56*56*256=800K	params: (3*3*256)*256 = 589,824
3x3 conv, 256	memory: 56*56*256=800K	params: (3*3*256)*256 = 589,824
Pool	memory: 28*28*256=200K	params: 0
3x3 conv, 512	memory: 28*28*512=400K	params: (3*3*256)*512 = 1,179,648
3x3 conv, 512	memory: 28*28*512=400K	params: (3*3*512)*512 = 2,359,296
3x3 conv, 512	memory: 28*28*512=400K	params: (3*3*512)*512 = 2,359,296
Pool	memory: 14*14*512=100K	params: 0
3x3 conv, 512	memory: 14*14*512=100K	params: (3*3*512)*512 = 2,359,296
3x3 conv, 512	memory: 14*14*512=100K	params: (3*3*512)*512 = 2,359,296
3x3 conv, 512	memory: 14*14*512=100K	params: (3*3*512)*512 = 2,359,296
Pool	memory: 7*7*512=25K	params: 0
FC 4096	memory: 4096	params: 7*7*512*4096 = 102,760,448
FC 4096	memory: 4096	params: 4096*4096 = 16,777,216
FC 1000	memory: 1000	params: 4096*1000 = 4,096,000

Carnegie Mellon University

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[Simonyan and Zisserman, 2014]

VGGNet

DETAILS/RETROSPECTIVES :

- ILSVRC'14 2ND IN CLASSIFICATION, 1ST IN LOCALIZATION
- SIMILAR TRAINING PROCEDURE AS ALEXNET
- NO LOCAL RESPONSE NORMALISATION (LRN)
- USE VGG16 OR VGG19 (VGG19 ONLY SLIGHTLY BETTER, MORE MEMORY)
- USE ENSEMBLES FOR BEST RESULTS
- FC7 FEATURES GENERALIZE WELL TO OTHER TASKS
- TRAINED ON 4 NVIDIA TITAN BLACK GPUS FOR **TWO TO THREE WEEKS.**

Carnegie Mellon University

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[Simonyan and Zisserman, 2014]

VGGNet

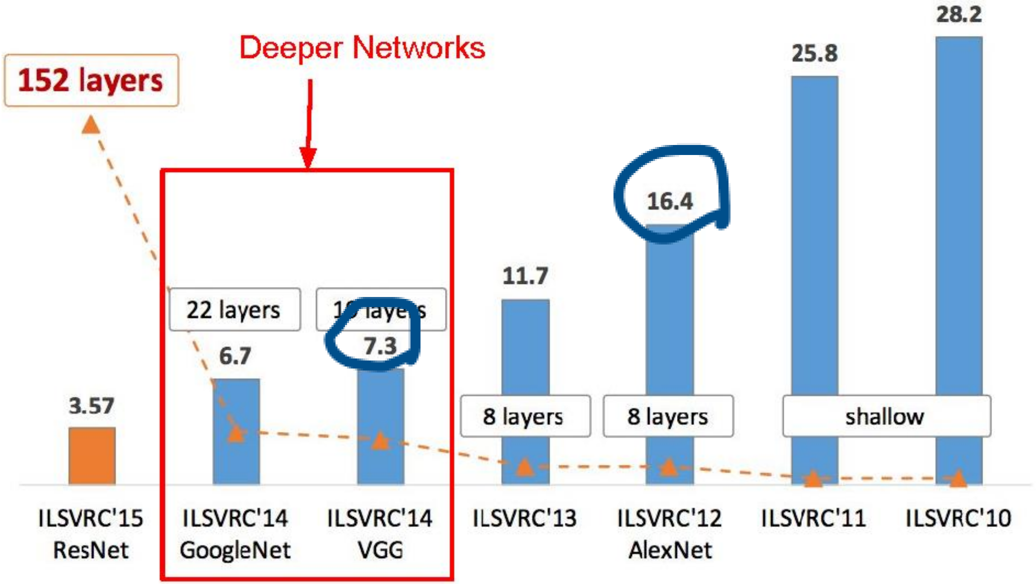
VGG Net reinforced the notion that **convolutional neural networks have to have a deep network of layers in order for this hierarchical representation of visual data to work.**

Keep it deep.

Keep it simple.

[Simonyan and Zisserman, 2014]

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners



Carnegie Mellon University

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

GoogleNet

Going Deeper with Convolutions - Christian Szegedy et al.; 2015

ILSVRC 2014 competition winner

Also significantly deeper than AlexNet

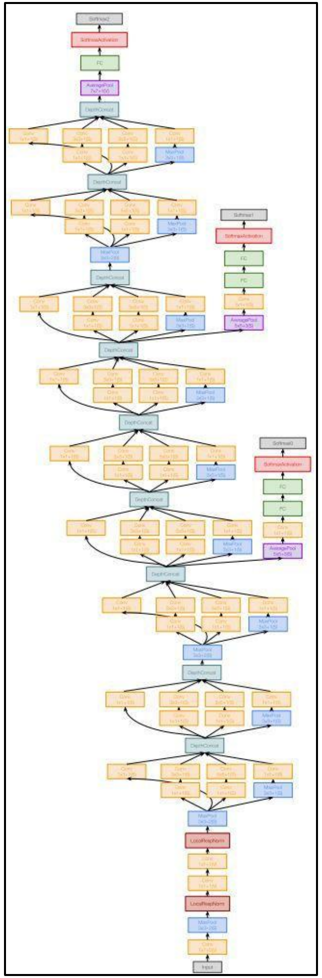
x12 less parameters than AlexNet

Focused on computational efficiency

Carnegie Mellon University

[Szegedy et al., 2014]

GoogleNet



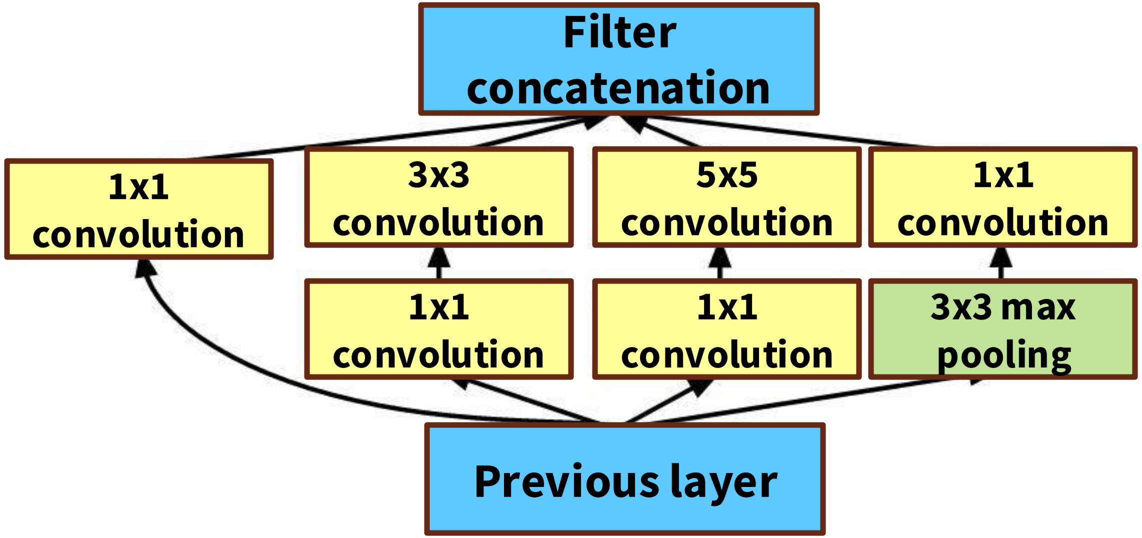
- 22 layers
- Efficient "**Inception**" **module** - strayed from the general approach of simply stacking conv and pooling layers on top of each other in a sequential structure
- No FC layers
- Only 5 million parameters!
- ILSVRC'14 classification winner (6.7% top 5 error)

Carnegie Mellon University

[Szegedy et al., 2014]

GoogleNet

“**INCEPTION MODULE**”: DESIGN A GOOD LOCAL NETWORK TOPOLOGY (NETWORK WITHIN A NETWORK) AND THEN STACK THESE MODULES ON TOP OF EACH OTHER

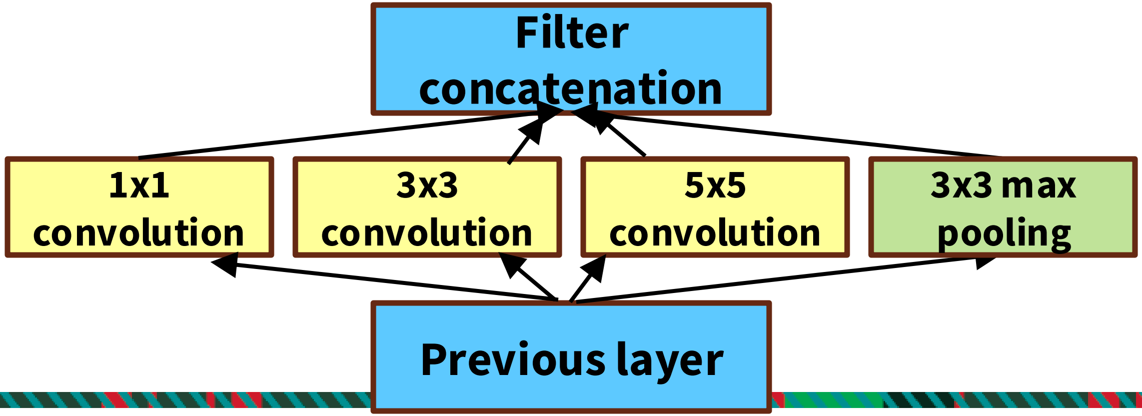


[Szegedy et al., 2014]

GoogleNet

Naïve Inception Model

- Apply parallel filter operations on the input :
  - Multiple receptive field sizes for convolution (1x1, 3x3, 5x5)
  - Pooling operation (3x3)
- Concatenate all filter outputs together depth-wise



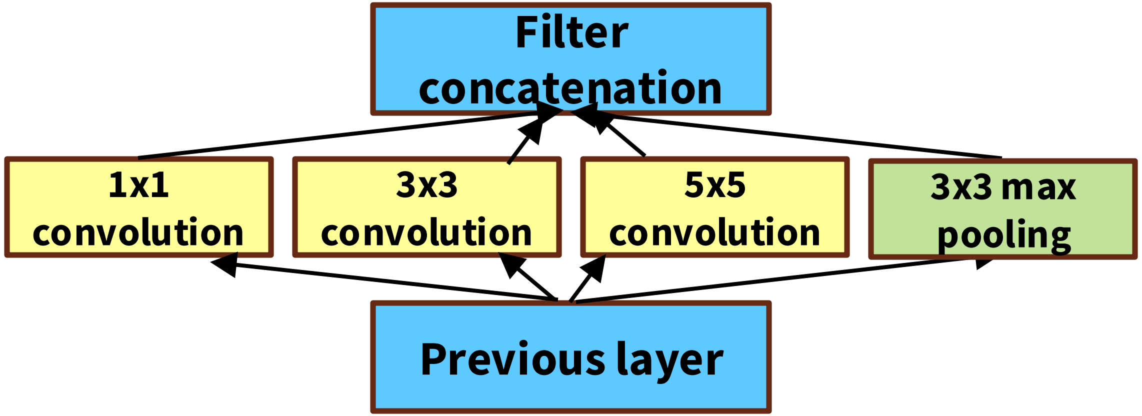
Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

Carnegie Mellon University

[Szegedy et al., 2014]

GoogleNet

- **What's the problem with this?**
  High computational complexity



Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[Szegedy et al., 2014]

GoogleNet

OUTPUT VOLUME SIZES:

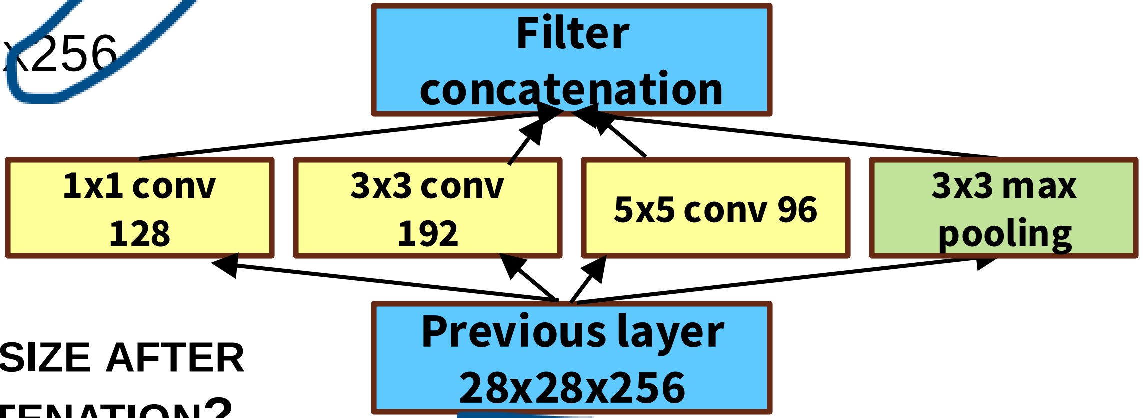
1x1 CONV, 128: 28x28x128

3x3 CONV, 192: 28x28x192

5x5 CONV, 96: 28x28x96

3x3 POOL: 28x28x256

Example:



WHAT IS OUTPUT SIZE AFTER FILTER CONCATENATION?

28x28x(128+192+96+256) = 28x28x672

[Szegedy et al., 2014]

GoogleNet

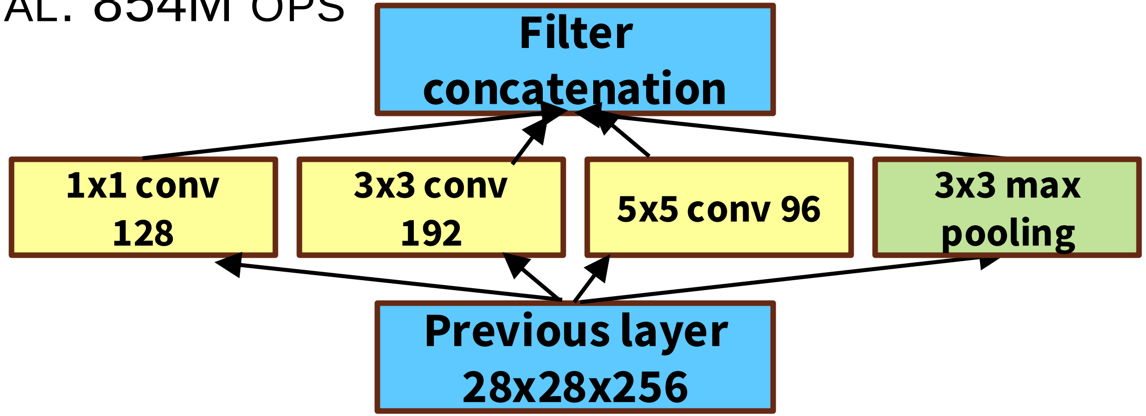
NUMBER OF CONVOLUTION OPERATIONS:

1X1 CONV, 128: 28X28X128X1X1X256

3X3 CONV, 192: 28X28X192X3X3X256

5X5 CONV, 96: 28X28X96X5X5X256

TOTAL: 854M OPS

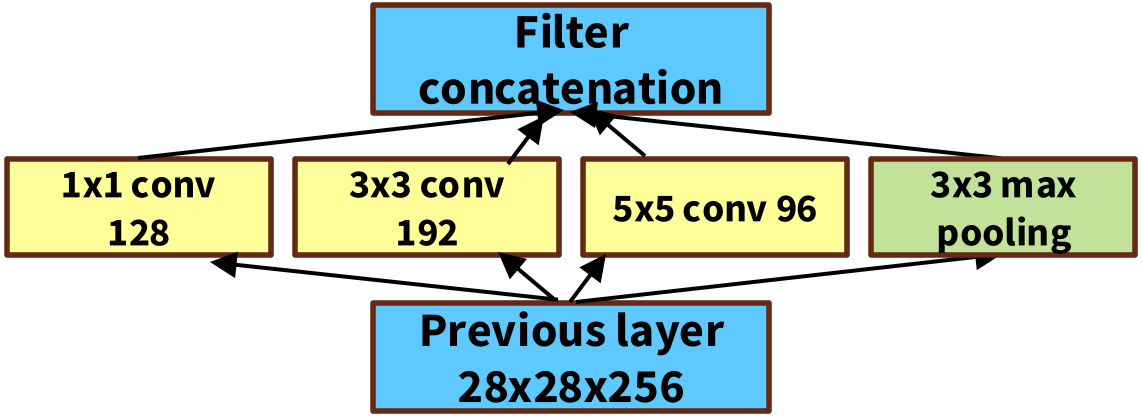


[Szegedy et al., 2014]

GoogleNet

VERY EXPENSIVE COMPUTE!

POOLING LAYER ALSO PRESERVES FEATURE DEPTH, WHICH MEANS TOTAL DEPTH AFTER CONCATENATION CAN ONLY GROW AT EVERY LAYER.

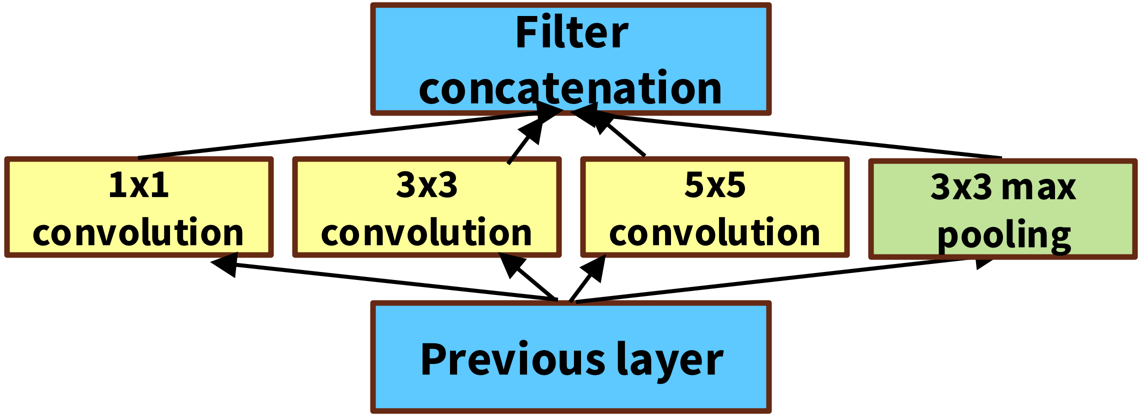


Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[Szegedy et al., 2014]

GoogleNet

SOLUTION: "BOTTLENECK" LAYERS THAT USE 1X1 CONVOLUTIONS TO REDUCE FEATURE DEPTH (FROM PREVIOUS HOUR).



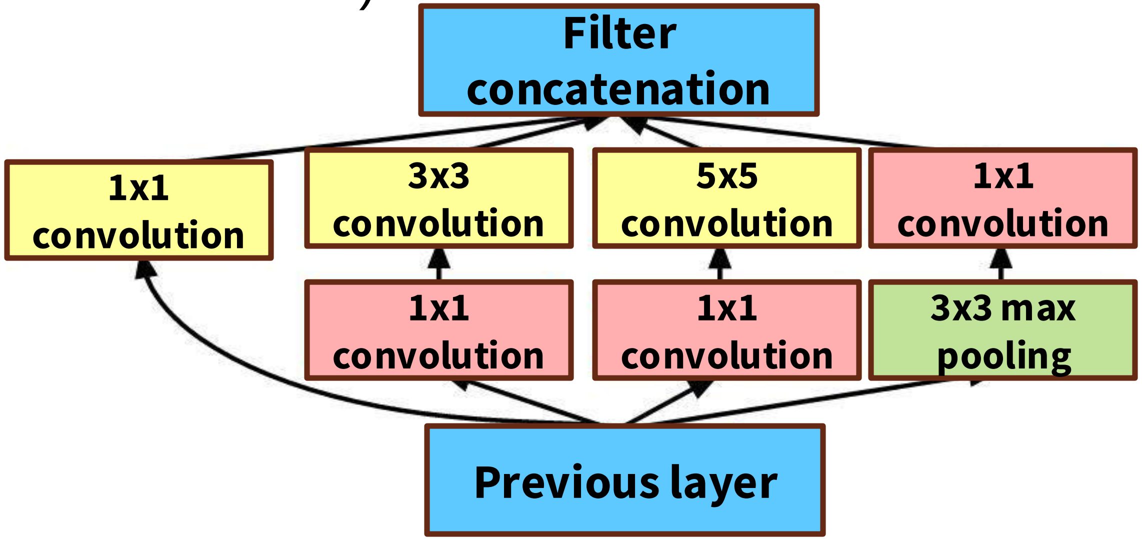
Carnegie Mellon University

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[Szegedy et al., 2014]

GoogleNet

SOLUTION: "BOTTLENECK" LAYERS THAT USE 1X1 CONVOLUTIONS TO REDUCE FEATURE DEPTH (FROM PREVIOUS HOUR).



Carnegie Mellon University

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[Szegedy et al., 2014]

Number of convolution operations:

1x1 conv, 64: 28x28x64x1x1x256

1x1 conv, 64: 28x28x64x1x1x256

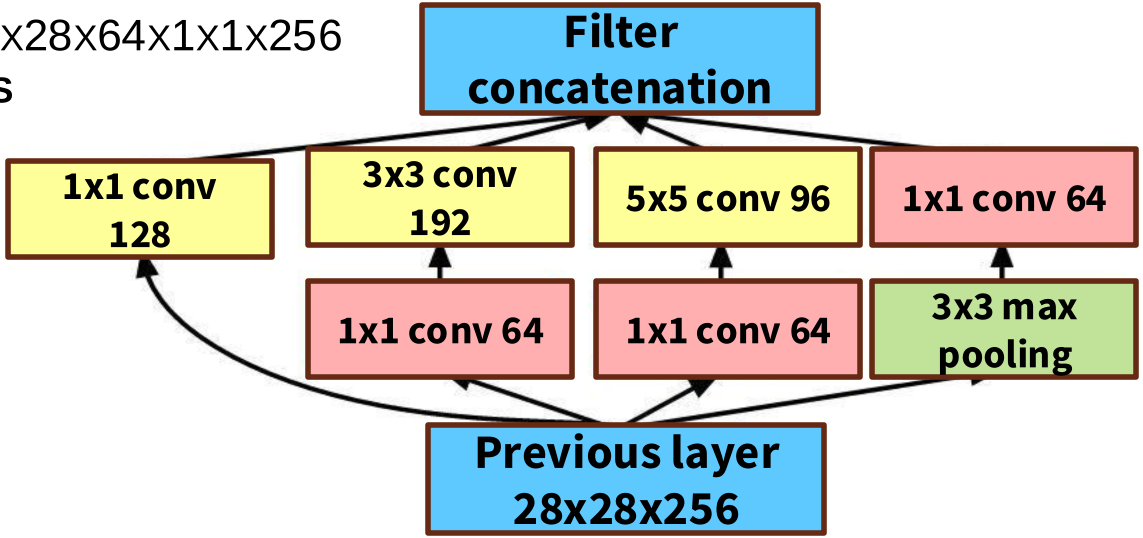
1x1 conv, 128: 28x28x128x1x1x256

3x3 conv, 192: 28x28x192x3x3x64

5x5 conv, 96: 28x28x96x5x5x264

1x1 conv, 64: 28x28x64x1x1x256

Total: 353M ops

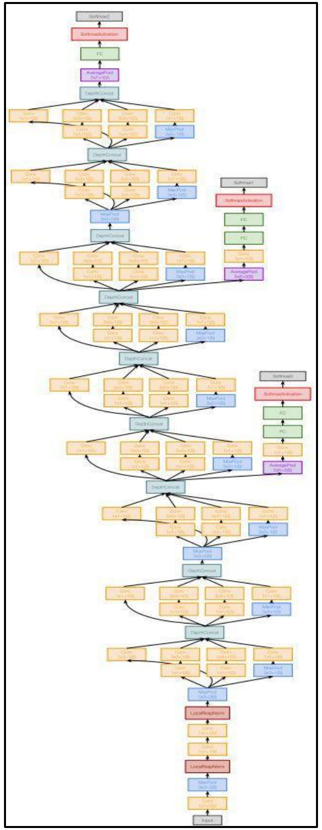


Compared to 854M ops for naive version

Carnegie Mellon University

[Szegedy et al., 2014]

GoogleNet



Details/Retrospectives :

- Deeper networks, with computational efficiency
- 22 layers
- Efficient “Inception” module
- No FC layers
- 12x less params than AlexNet
- ILSVRC’14 classification winner (6.7% top 5 error)

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[Szegedy et al., 2014]

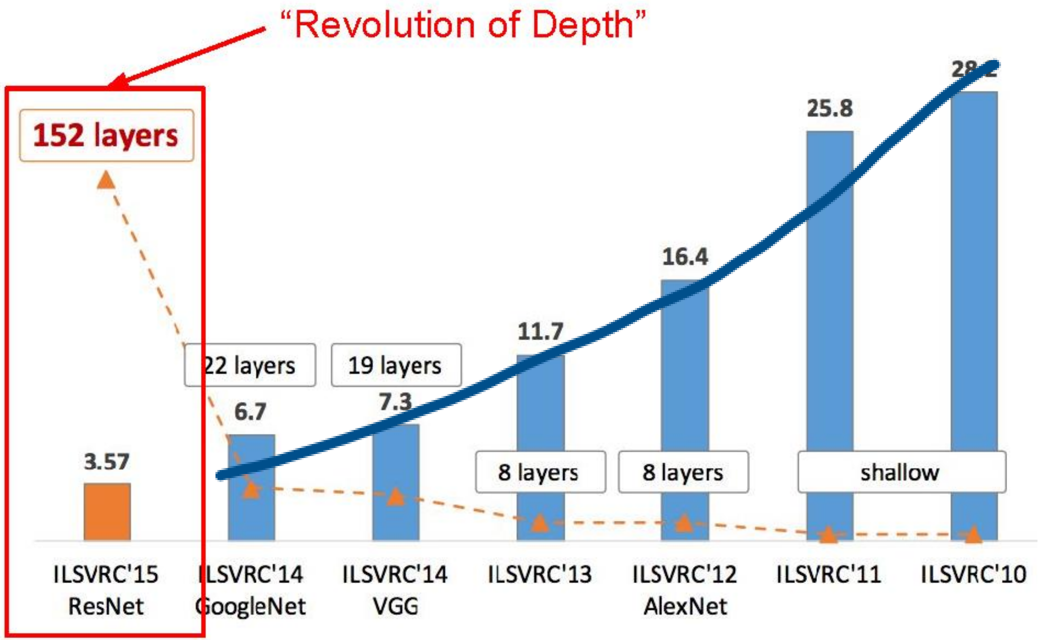
GoogleNet

INTRODUCED THE IDEA THAT CNN LAYERS **DIDN'T ALWAYS HAVE TO BE STACKED UP SEQUENTIALLY.** COMING UP WITH THE INCEPTION MODULE, THE AUTHORS SHOWED THAT A CREATIVE STRUCTURING OF LAYERS CAN LEAD TO IMPROVED PERFORMANCE AND **COMPUTATIONALLY EFFICIENCY**.

Carnegie Mellon University

[Szegedy et al., 2014]

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners



Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

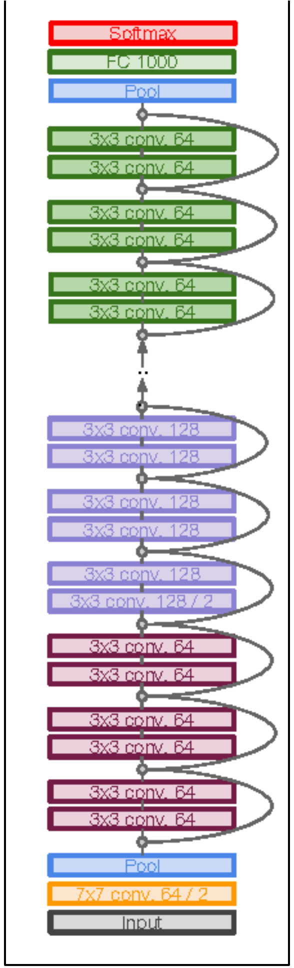
ResNet

- *Deep Residual Learning for Image Recognition - Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; 2015*
- Extremely deep network – 152 layers
- Deeper neural networks are more difficult to train.
- Deep networks suffer from vanishing and exploding gradients.
- Present a residual learning framework to ease the training of networks that are substantially deeper than those used previously.

Carnegie Mellon University

[He et al., 2015]

ResNet



- ILSVRC'15 classification winner (3.57% top 5 error, humans generally hover around a 5-10% error rate)
  Swept all classification and detection competitions in ILSVRC'15 and COCO'15!

Carnegie Mellon University

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[He et al., 2015]

ResNet

- What happens when we continue stacking deeper layers on a convolutional neural network?

- 56-layer model performs worse on both training and test error

-> The deeper model performs worse (not caused by overfitting)!

Carnegie Mellon University

[He et al., 2015]

ResNet

- **Hypothesis**: The problem is an optimization problem. Very deep networks are harder to optimize.
- **Solution**: Use network layers to fit residual mapping instead of directly trying to fit a desired underlying mapping.

- We will use **skip connections** allowing us to take the activation from one layer and feed it into another layer, much deeper into the network.
- Use layers to fit residual $F(x) = H(x) - x$ instead of $H(x)$ directly

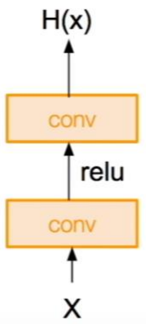
Carnegie Mellon University

[He et al., 2015]

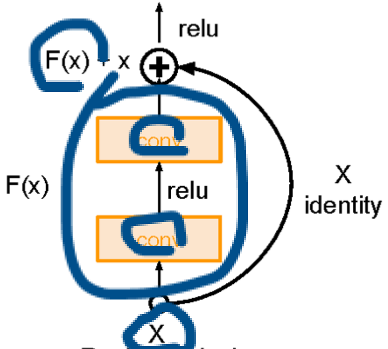
ResNet

RESIDUAL BLOCK

INPUT X GOES THROUGH CONV-RELU-CONV SERIES AND GIVES US F(X). THAT RESULT IS THEN ADDED TO THE ORIGINAL INPUT X. LET'S CALL THAT $\boxed{H(X) = F(X) + X.}$
IN TRADITIONAL CNNS, H(X) WOULD JUST BE EQUAL TO F(X).
SO, INSTEAD OF JUST COMPUTING THAT TRANSFORMATION (STRAIGHT FROM X TO F(X)), WE'RE COMPUTING THE TERM THAT WE HAVE TO *ADD*, F(X), TO THE INPUT, X.



"Plain" layers

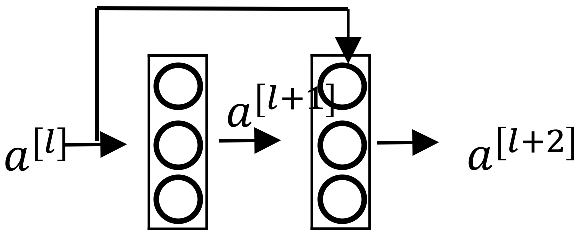


Residual block

Carnegie Mellon University

[He et al., 2015]

ResNet



Short cut/ skip connection

$a^{[l]} \rightarrow$ **Linear** $\rightarrow$ **ReLU** $\rightarrow$ **Linear** $\rightarrow$ **ReLU** $\rightarrow a^{[l+2]}$

$a^{[l+1]}$

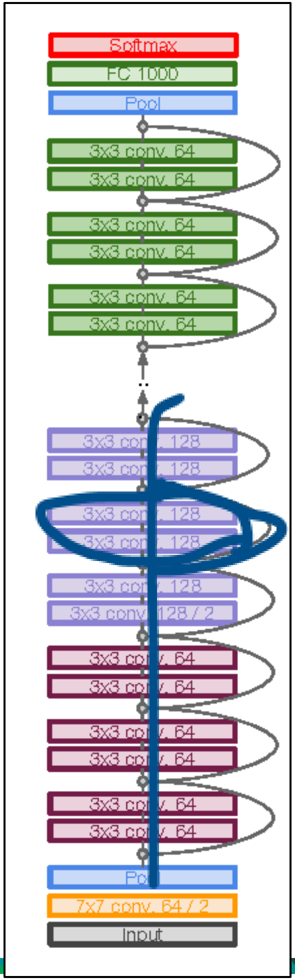
$$\mathbf{z^{[l+1]} = W^{[l+1]}\, a^{[l]} + b^{[l+1]}} \qquad \mathbf{z^{[l+2]} = W^{[l+2]} a^{[l+1]} + b^{[l+2]}}$$

$$\mathbf{a^{[l+1]} = g(z^{[l+1]})} \qquad \mathbf{a^{[l+2]} = g(z^{[l+2]})}$$

$$\mathbf{a^{[l+2]} = g\left(z^{[l+2]} + a^{[l]}\right) = g(W^{[l+2]} a^{[l+1]} + b^{[l+2]} + a^{[l]})}$$

[He et al., 2015]

ResNet



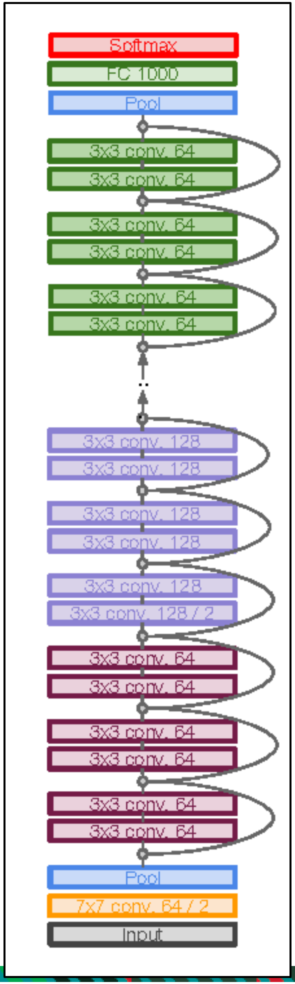
Full ResNet architecture:

- Stack residual blocks
- Every residual block has two 3x3 conv layers
- Periodically, double # of filters and downsample spatially using stride 2 (in each dimension)
- Additional conv layer at the beginning
- No FC layers at the end (only FC 1000 to output classes)

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[He et al., 2015]

ResNet



- Total depths of 34, 50, 101, or 152 layers for ImageNet
- For deeper networks (ResNet-50+), use "bottleneck" layer to improve efficiency (similar to GoogLeNet)

Carnegie Mellon University

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[He et al., 2015]

EXPERIMENTAL RESULTS:

ABLE TO TRAIN VERY DEEP NETWORKS WITHOUT DEGRADING

DEEPER NETWORKS NOW ACHIEVE LOWER TRAINING ERRORS
AS EXPECTED

[He et al., 2015]

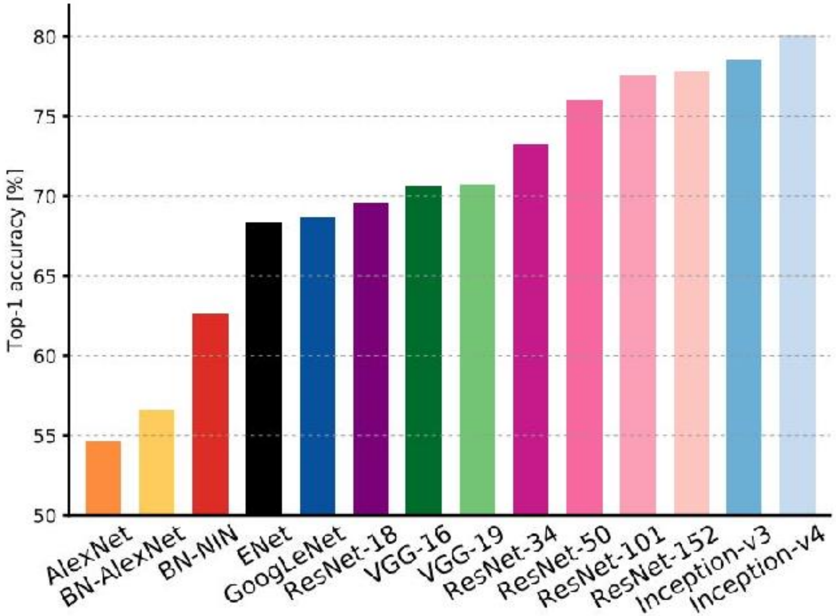
ResNet

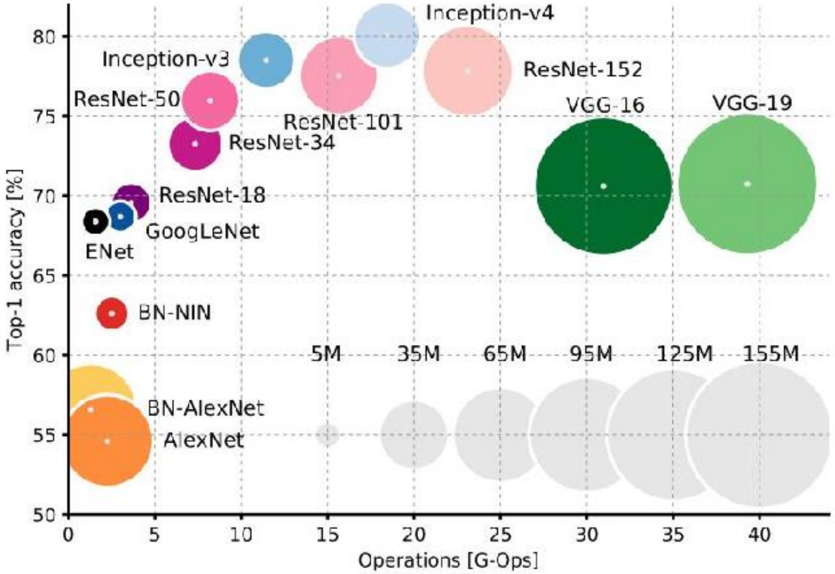
THE **BEST** CNN ARCHITECTURE THAT WE CURRENTLY HAVE AND IS A GREAT INNOVATION FOR THE IDEA OF RESIDUAL LEARNING.

EVEN BETTER THAN HUMAN PERFORMANCE!

[He et al., 2015]

Accuracy comparison

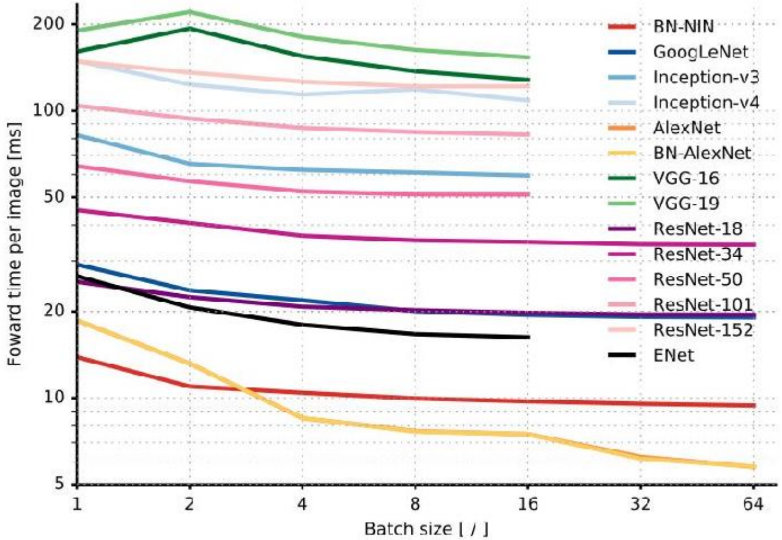


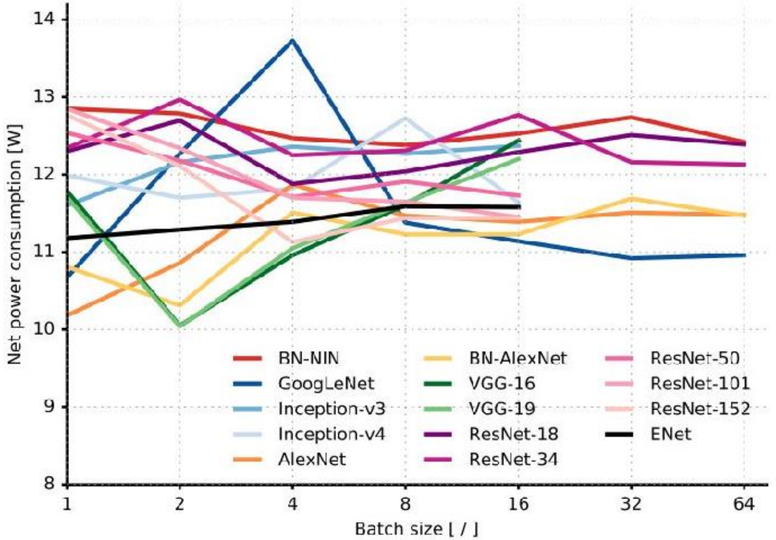


Carnegie Mellon University

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

Forward pass time and power consumption





Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

Summary

LeNet-5

AlexNet

VGG

GoogLeNet – Inception module

ResNet – Residual block

References

Gradient-based learning applied to document recognition; Ann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner; 1998

ImageNet Classification with Deep Convolutional Neural Networks - Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton; 2012

Very Deep Convolutional Networks For Large Scale Image Recognition - Karen Simonyan and Andrew Zisserman; 2015

Going Deeper with Convolutions - Christian Szegedy et al.; 2015

Deep Residual Learning for Image Recognition - Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; 2015

Stanford CS231- Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9

Coursera, Machine Learning course by Andrew Ng.

References

The 9 Deep Learning Papers You Need To Know About (Understanding CNNs Part 3) by Adit Deshpande https://adeshpande3.github.io/adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html

CNNs Architectures: LeNet, AlexNet, VGG, GoogLeNet, ResNet and more … By Siddharth Das https://medium.com/@siddharthdas_32104/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5

Slide taken from Forward And Backpropagation in Convolutional Neural Network. – Medium , By Sujit Rai https://medium.com/@2017csm1006/forward-and-backpropagation-in-convolutional-neural-network-4dfa96d7b37e