Advanced Section #5:
State Of The Art (SOTA)

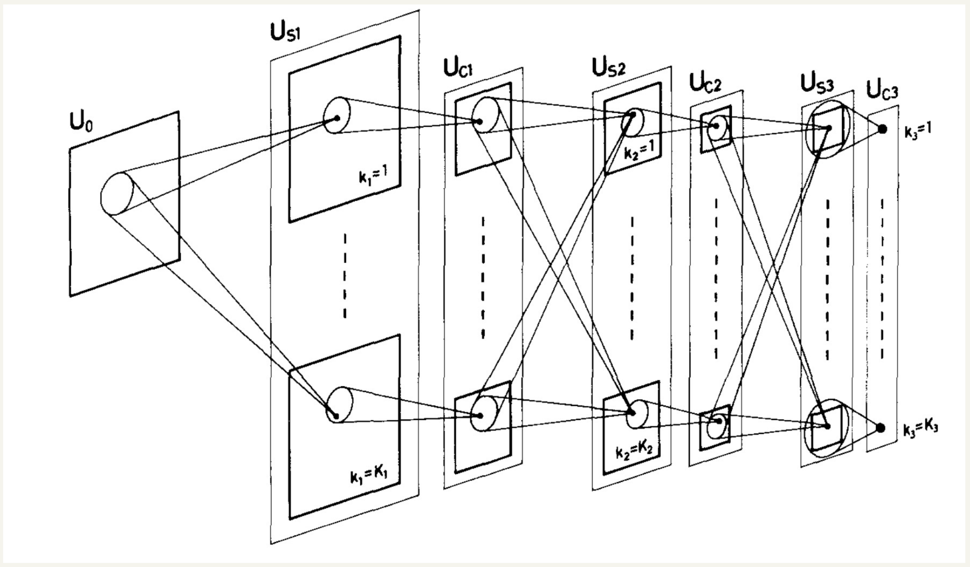
Pavlos Protopapas

CS109B Advanced Topics in Data Science

Pavlos Protopapas

Initial ideas

- The first research proposing something similar to a Convolutional Neural Network was authored by Kunihiko Fukushima in **1980** and was called the NeoCognitron.
- Inspired by discoveries on visual cortex of mammals.
- Fukushima applied the NeoCognitron to hand-written character recognition.

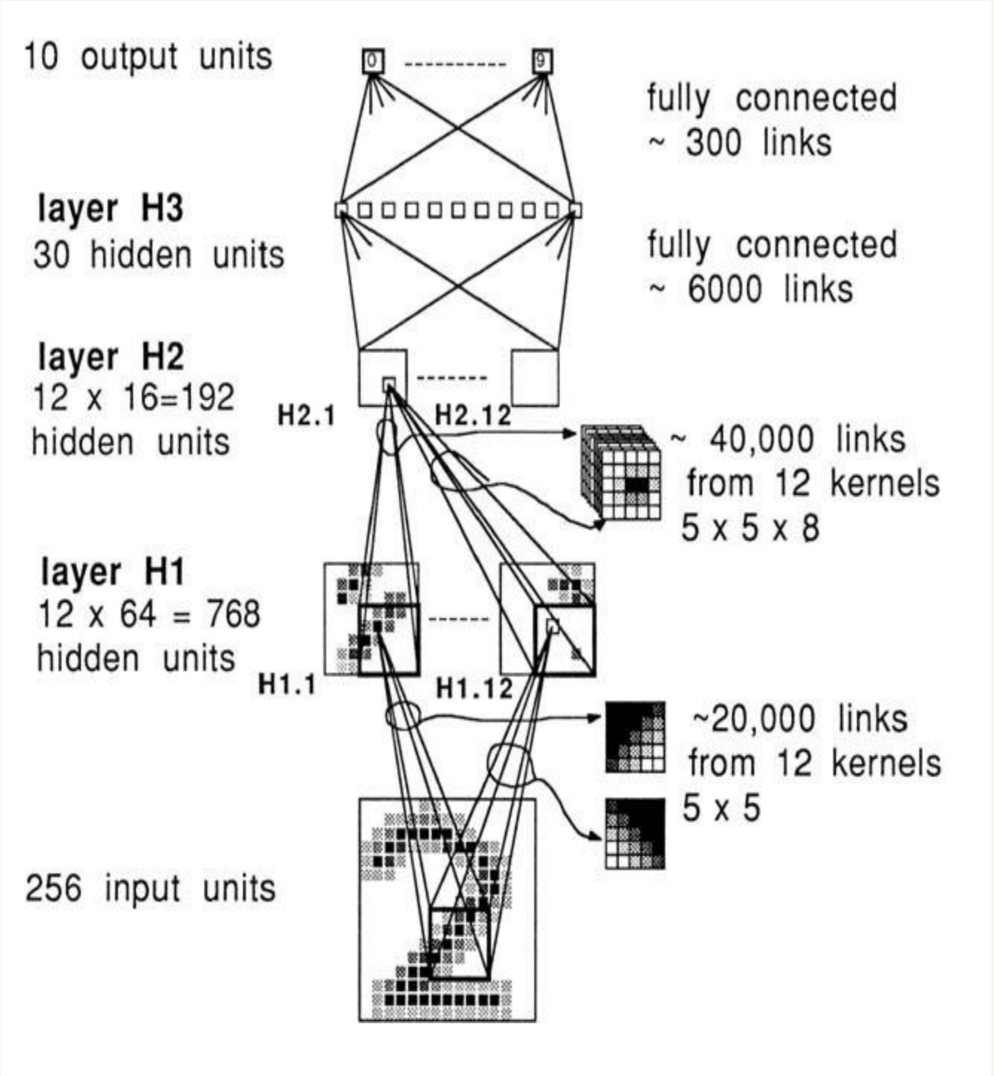


[1] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics, 36(4): 93-202, 1980.

Initial ideas

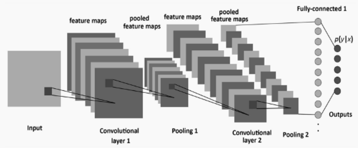
End of the 80's: several papers advanced the field

- Backpropagation published in French by Yann LeCun in 1985 (independently discovered by other researchers as well)
- TDNN by Waibel et al., 1989 - Convolutional-like network trained with backprop.
- Backpropagation applied to handwritten zip code recognition by LeCun et al., 1989

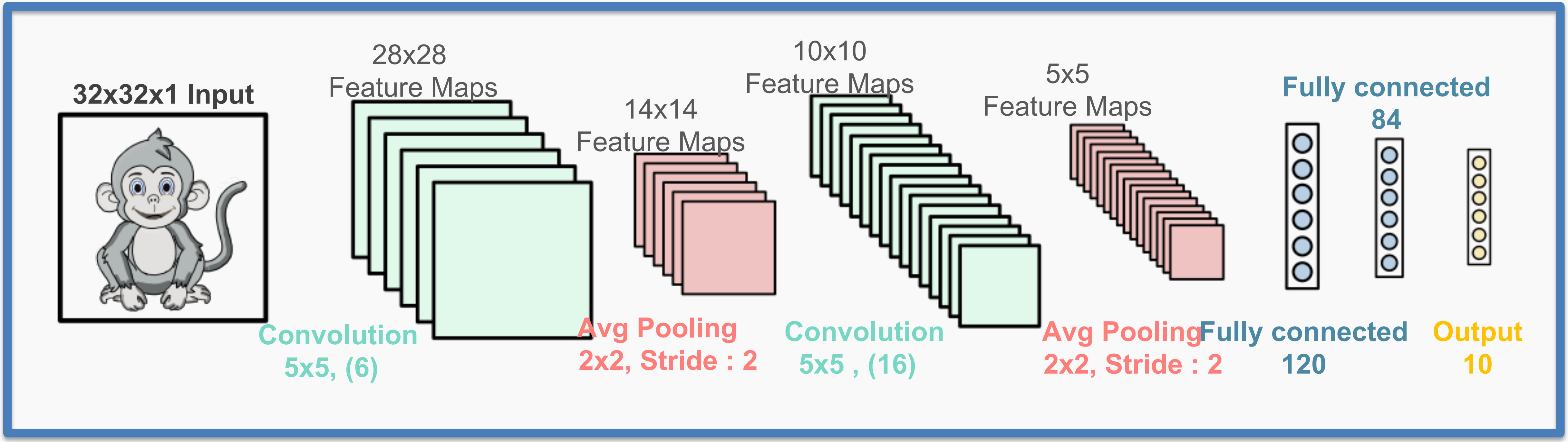


LeCun et al., 1989

LeNet



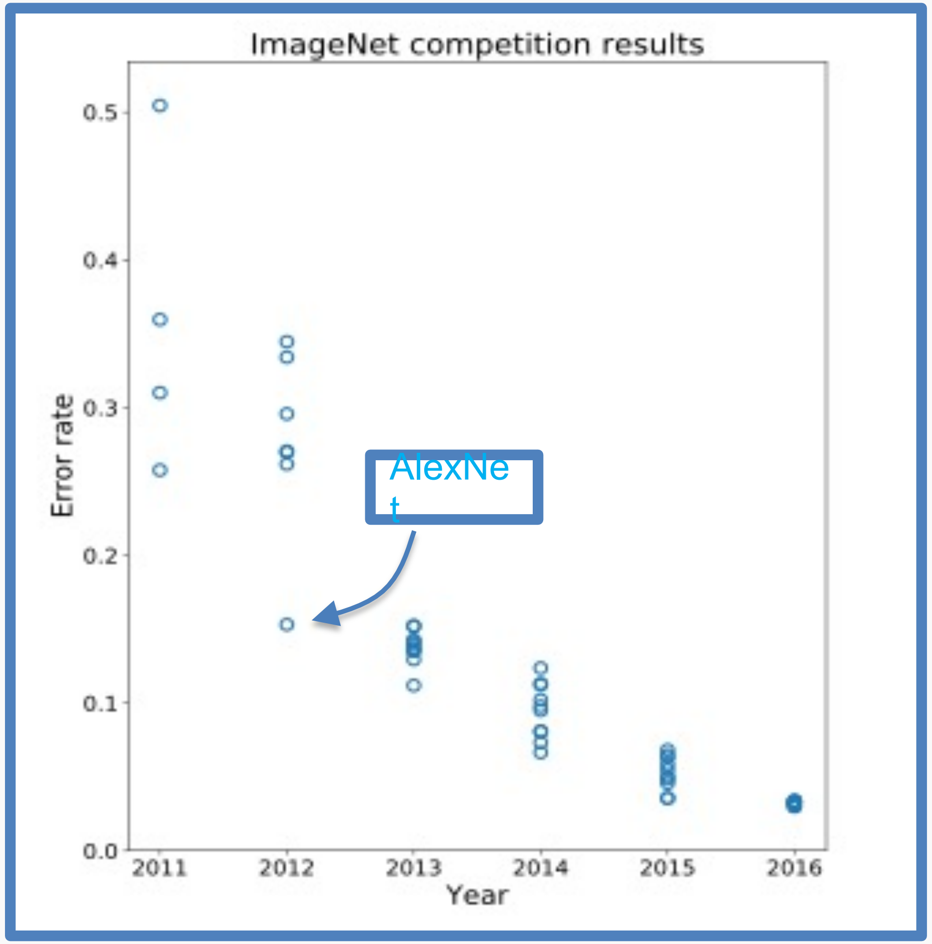
- November 1998: LeCun publishes one of his most recognized papers describing a “modern” CNN architecture for document recognition, called LeNet[1].
- Not his first iteration, this was in fact LeNet-5, but this paper is the commonly cited publication when talking about LeNet.



LeCun, Yann, et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86.11 (1998): 2278-2324.

AlexNet

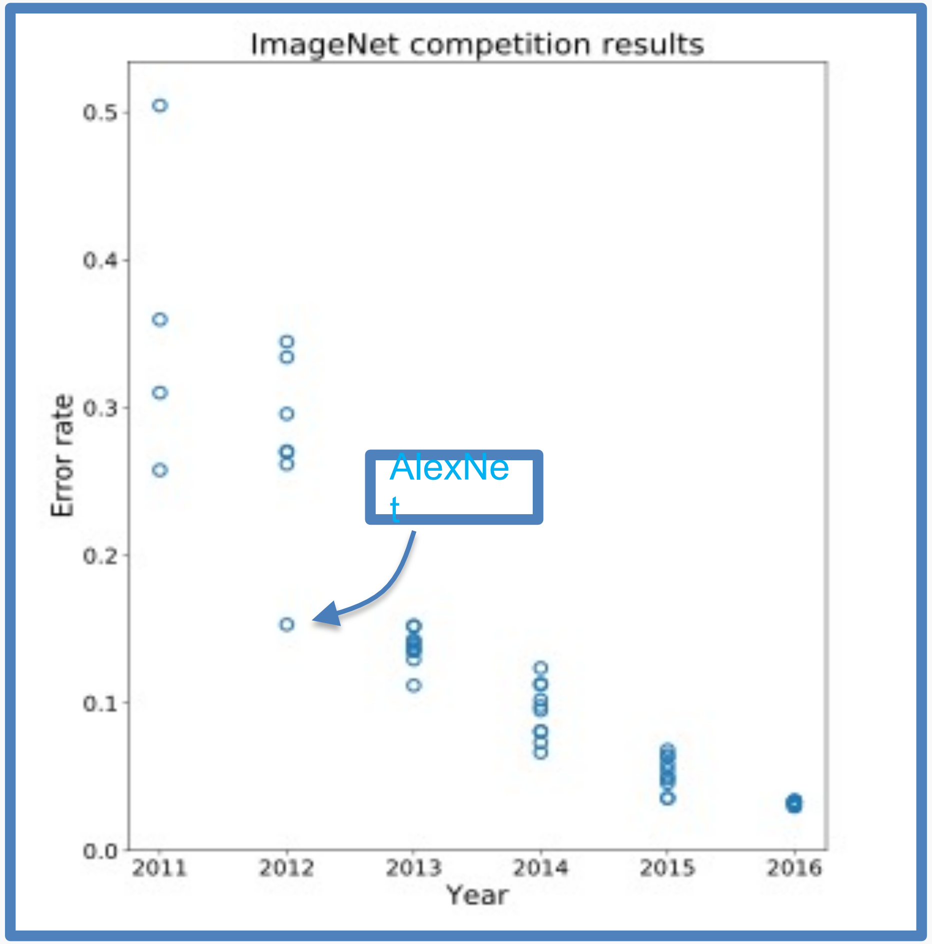
- Developed by Alex Krizhevsky, Ilya Sutskever and Geoffrey Hinton at Utoronto in 2012. More than 25000 citations.
- Destroyed the competition in the 2012 ImageNet Large Scale Visual Recognition Challenge. Showed benefits of CNNs and kickstarted AI revolution.
- top-5 error of 15.3%, more than 10.8 percentage points lower than runner-up.



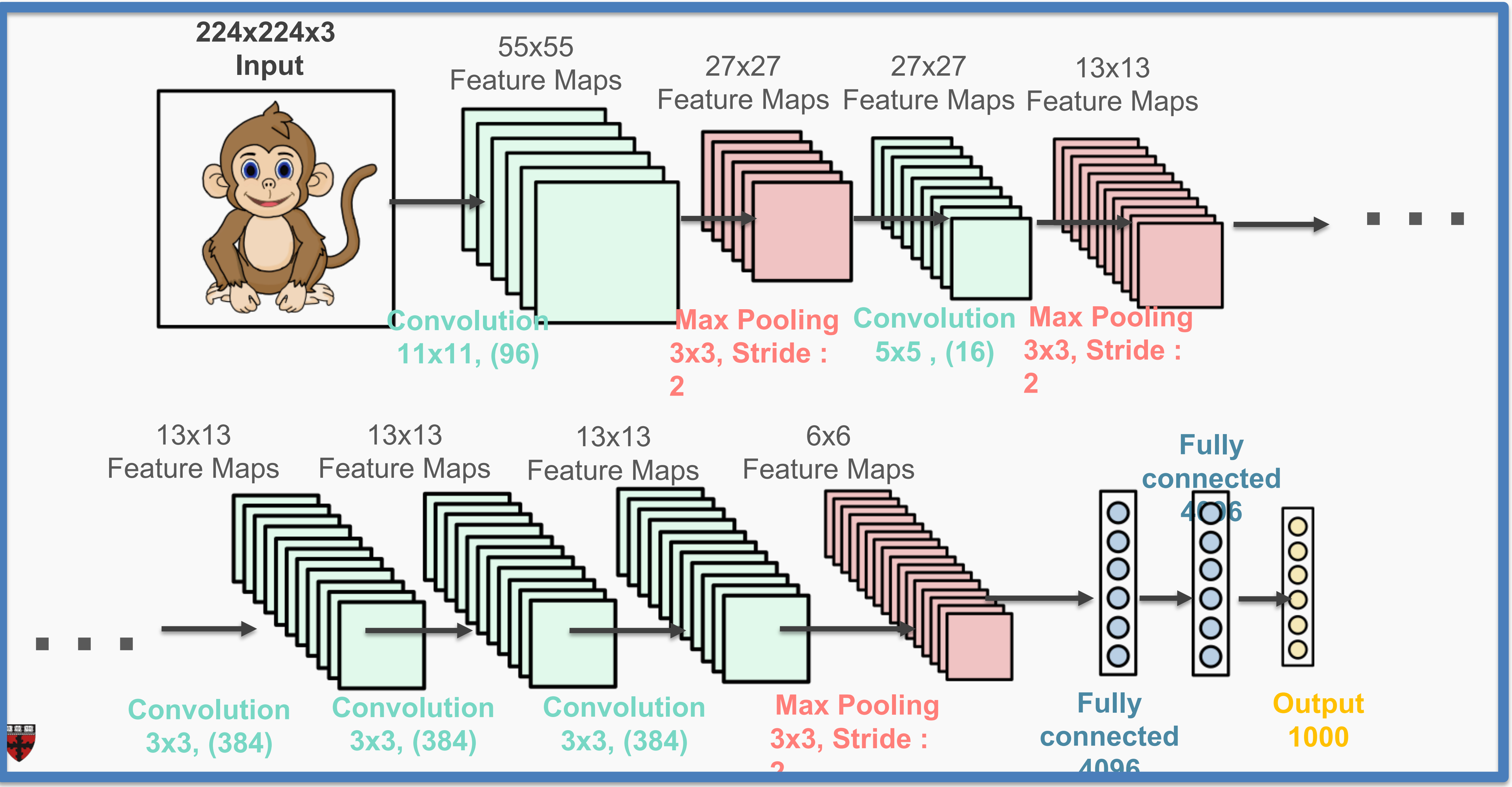
AlexNet

- Developed by Alex Krizhevsky, Ilya Sutskever and Geoffrey Hinton at Utoronto in 2012. More than 25000 citations.
- Destroyed the competition in the 2012 ImageNet Large Scale Visual Recognition Challenge. Showed benefits of CNNs and kickstarted AI revolution.
- top-5 error of 15.3%, more than 10.8 percentage points lower than runner-up.

- Main contributions:
  - Trained on ImageNet with data augmentation.
  - Increased depth of model, GPU training (*six days*).
  - Smart optimizer and Dropout layers.
  - ReLU activation!



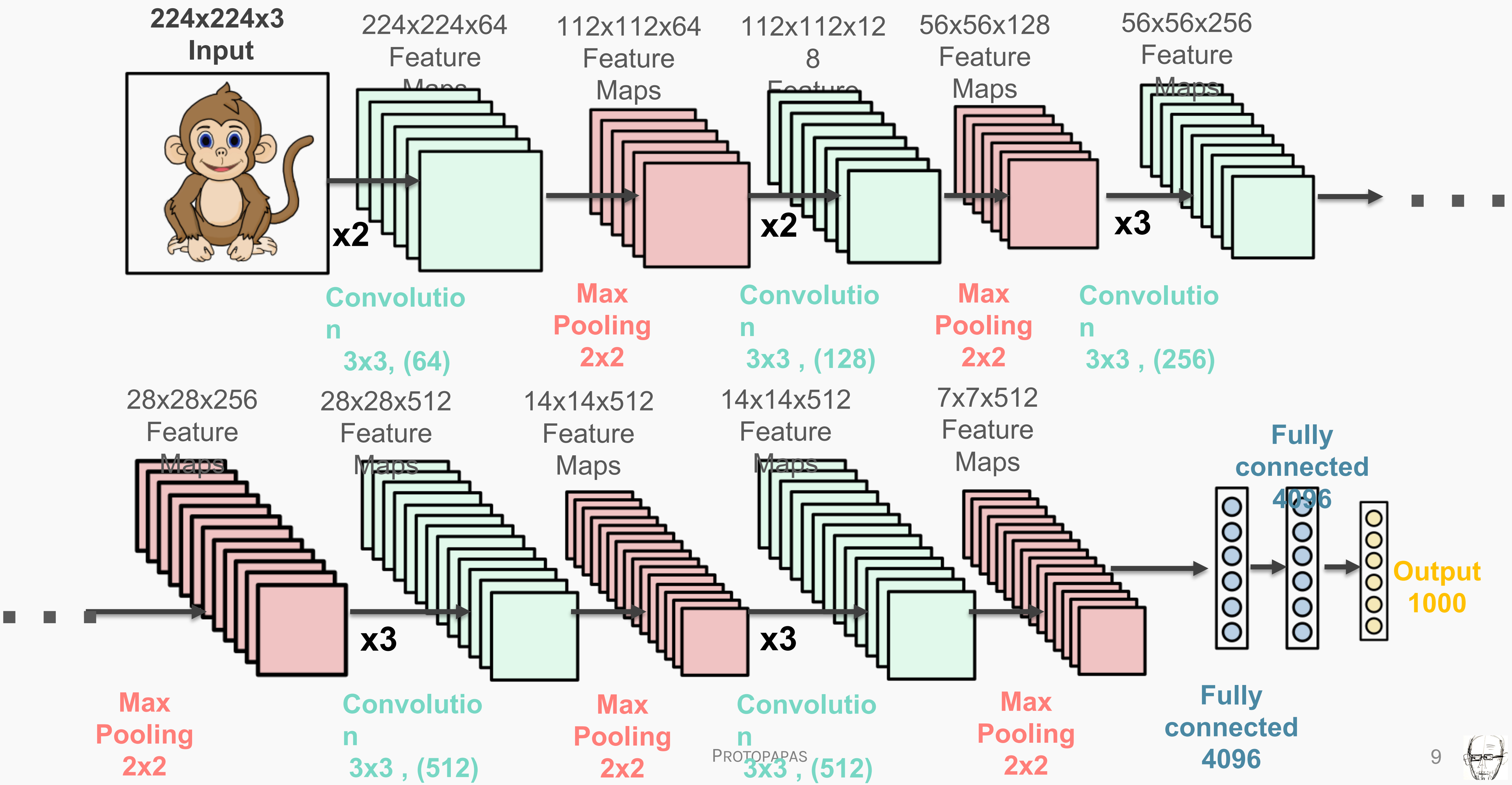
- 1.2 million high-resolution (227x227x3) images in the ImageNet 2010 contest
- 1000 different classes, NN with 60 million parameters to optimize (~ 255 MB)
- Uses ReLu activation functions; GPUs for training, 12 layers



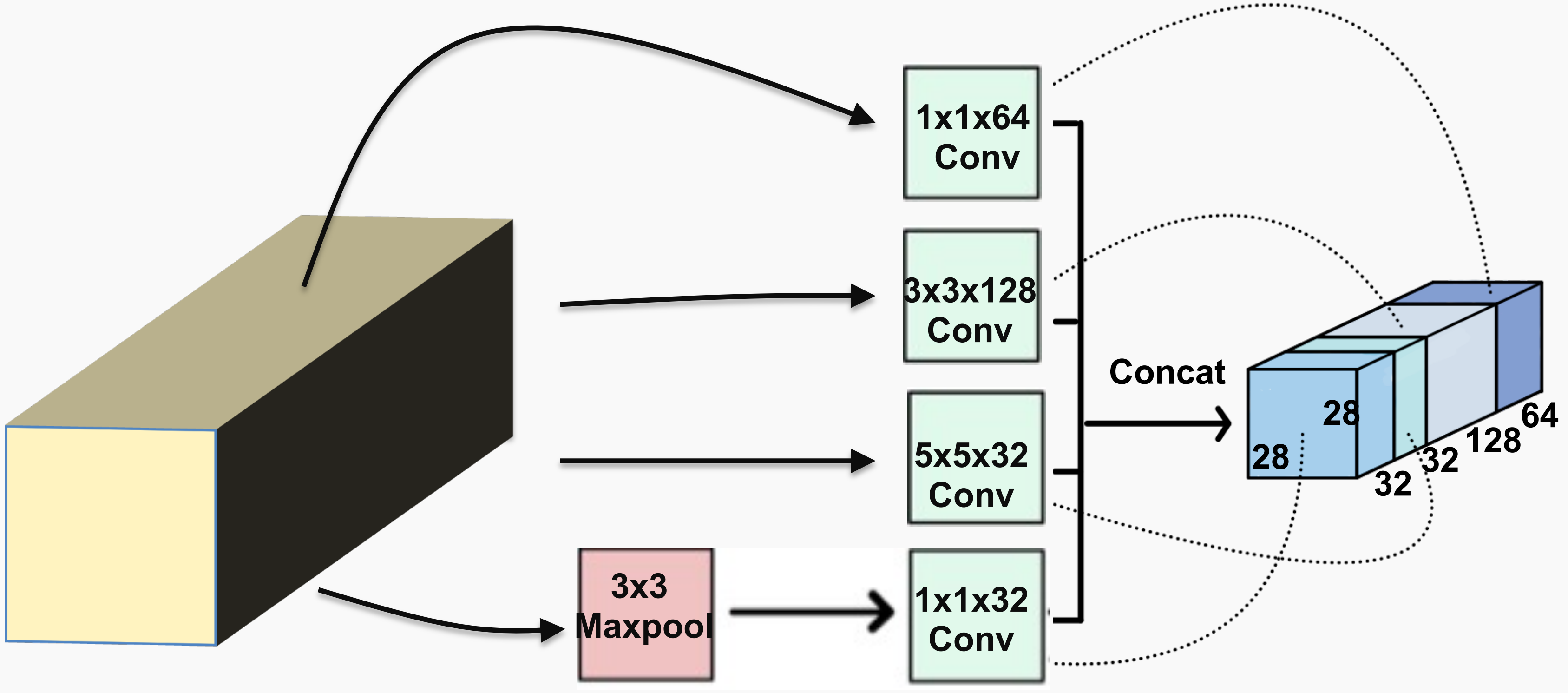
VGG

- Introduced by Simonyan and Zisserman (Oxford) in 2014.
- Simplicity and depth as main points. Used **3x3 filters exclusively** and 2x2 MaxPool layers with stride 2.
- Showed that two 3x3 filters have an effective receptive field of 5x5.
- As spatial size decreases, depth increases.
- Convolutional layers use ‘same’ padding and stride s=1.
- Max-pooling layers use a window size 2 and stride s=2.

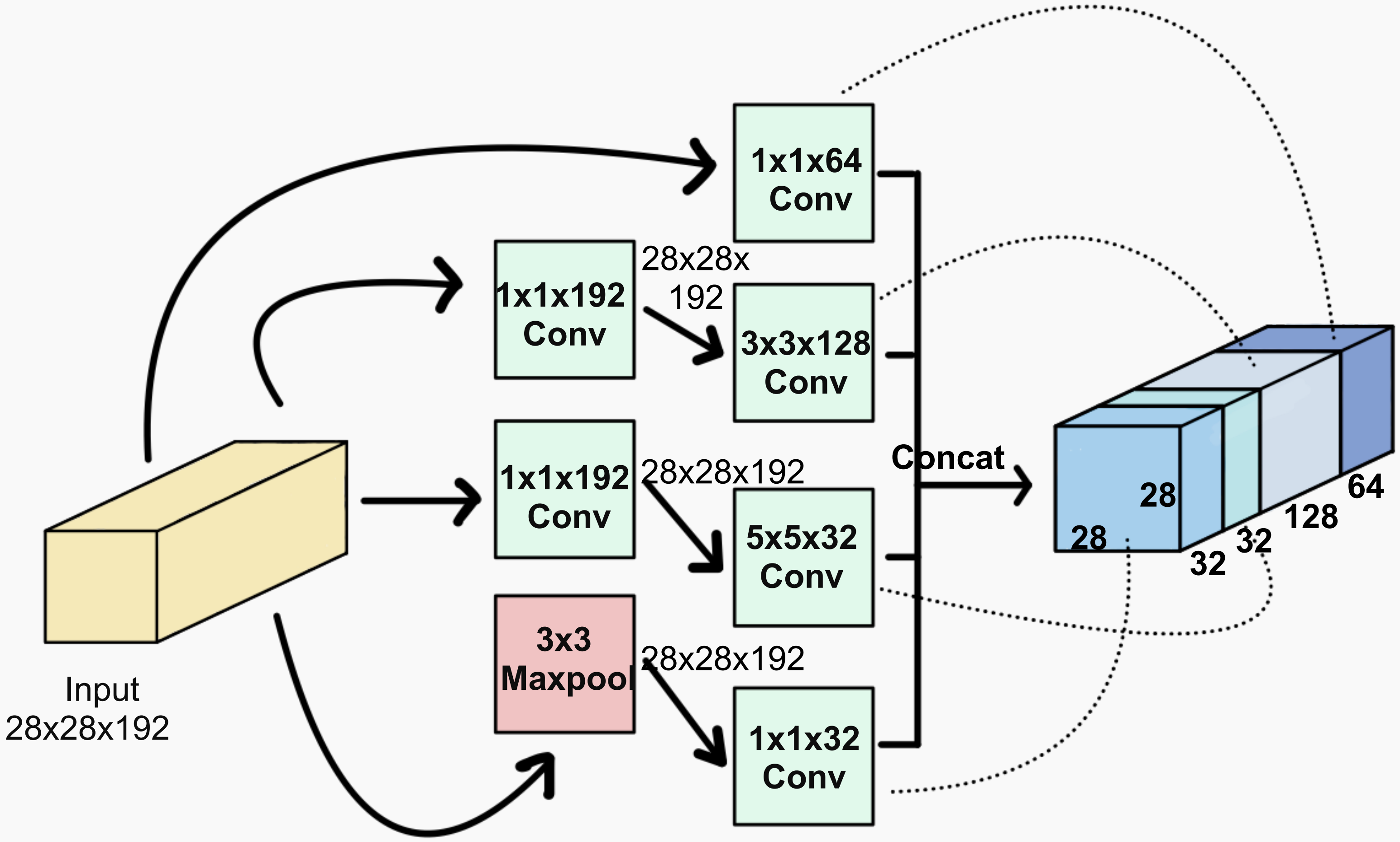
- ImageNet Challenge 2014; 16 or 19 layers; 138 million parameters.
- Trained for two to three weeks.
- Still used as of today.


224x224x3
Input
224x224x64
Feature
Maps
112x112x64
Feature
Maps
112x112x12
8
Feature
56x56x128
Feature
Maps
56x56x256
Feature
Maps
x2
x2
x3
Convolutio
n
3x3, (64)
Max
Pooling
2x2
Convolutio
n
3x3 , (128)
Max
Pooling
2x2
Convolutio
n
3x3 , (256)
28x28x256
Feature
Maps
28x28x512
Feature
Maps
14x14x512
Feature
Maps
14x14x512
Feature
Maps
7x7x512
Feature
Maps
Fully
connected
4096
x3
x3
Output
1000
Max
Pooling
2x2
Convolutio
n
3x3 , (512)
Max
Pooling
2x2
Convolutio
n
3x3 , (512)
Max
Pooling
2x2
Fully
connected
4096

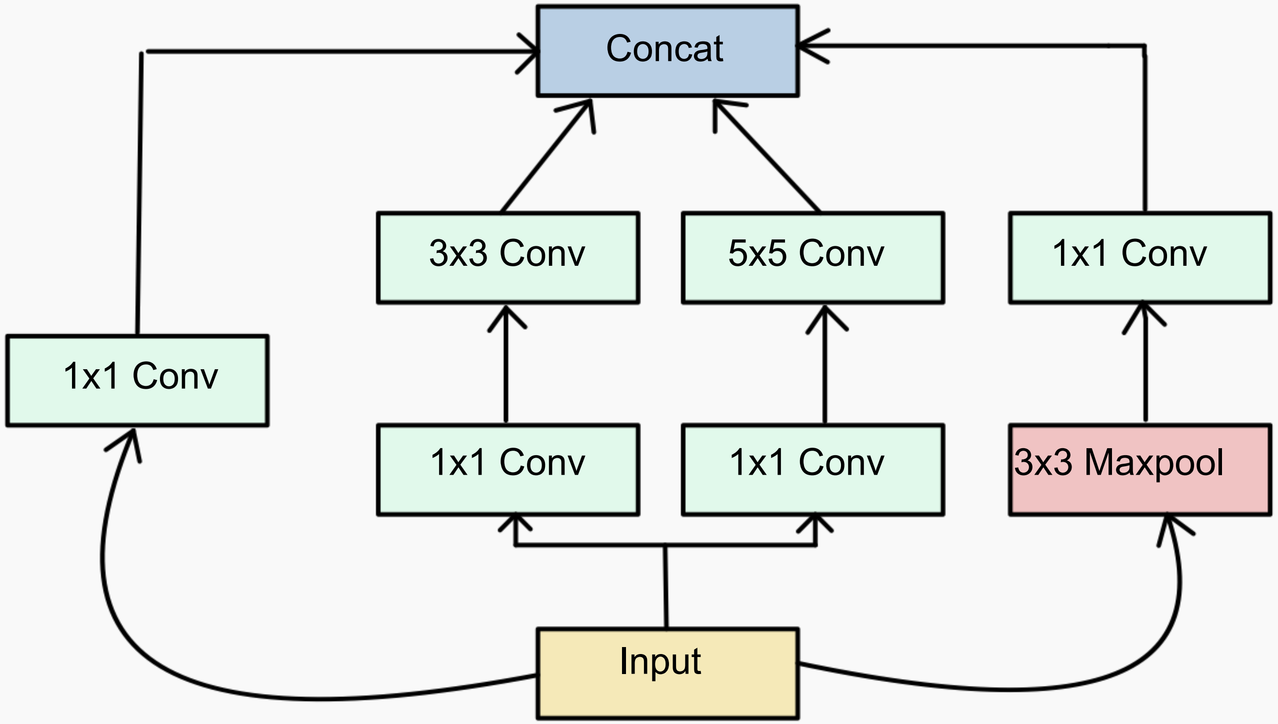
- The motivation behind inception networks is to use more than a singe type of convolution layer at each layer.
- Use `1x1,3x3,5x5` convolutional layers, and max-pooling layers in parallel.
- All modules use `same` convolution.



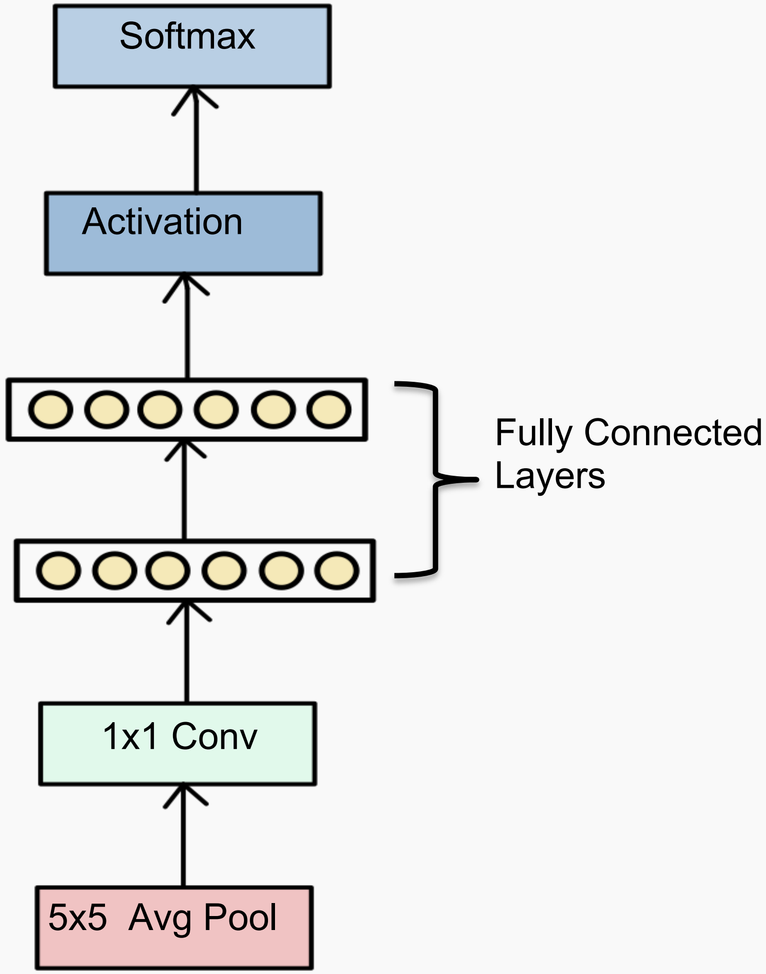
- Use 1 x 1 convolutions that reduce the size of the channel dimension.
    - The number of channels can vary from the input to the output.



The Inception Block

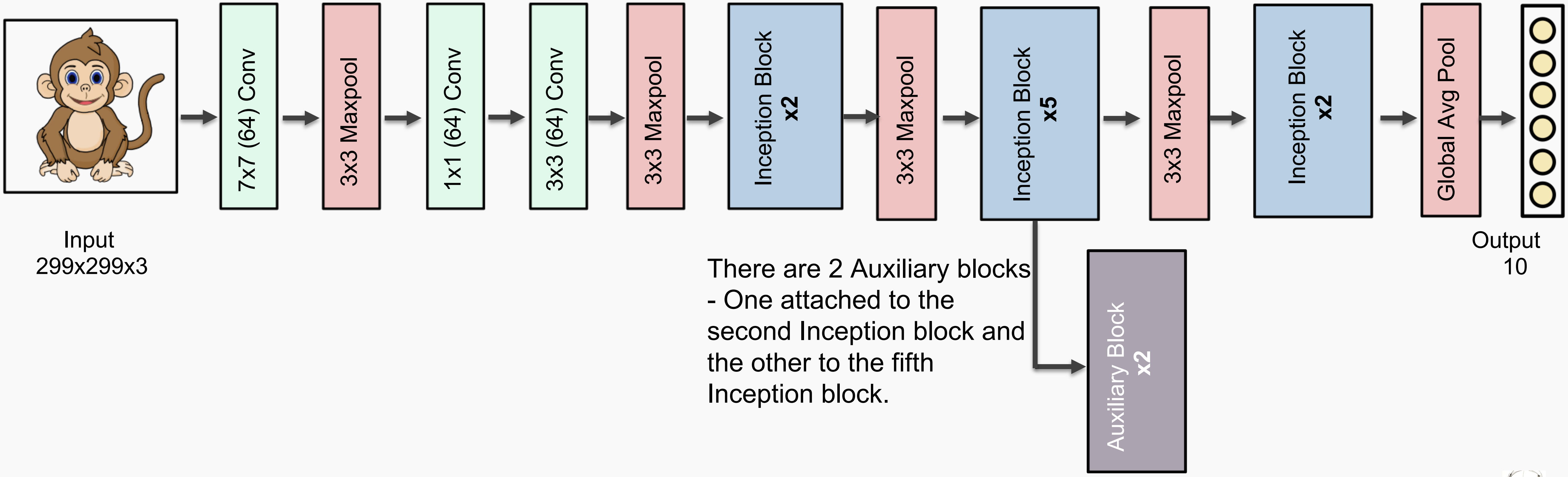


Auxiliary Block



SOTA Deep Models: Inception (GoogLeNet)

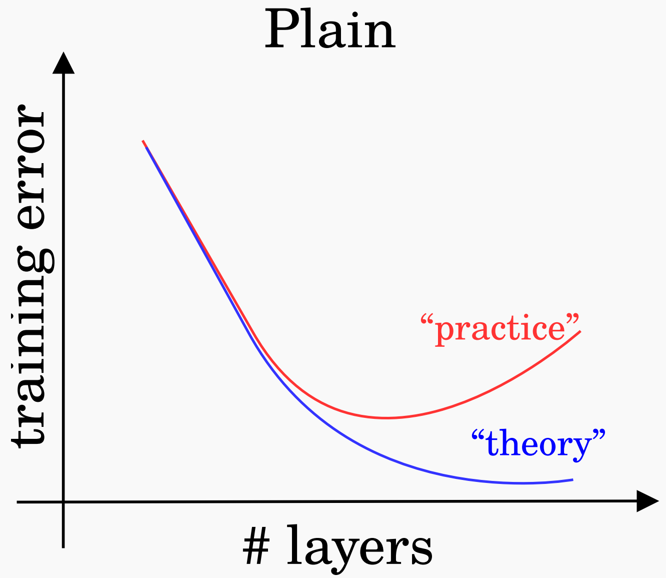
- The inception network is formed by concatenating other inception modules.
- It includes several softmax output units to enforce regularization.

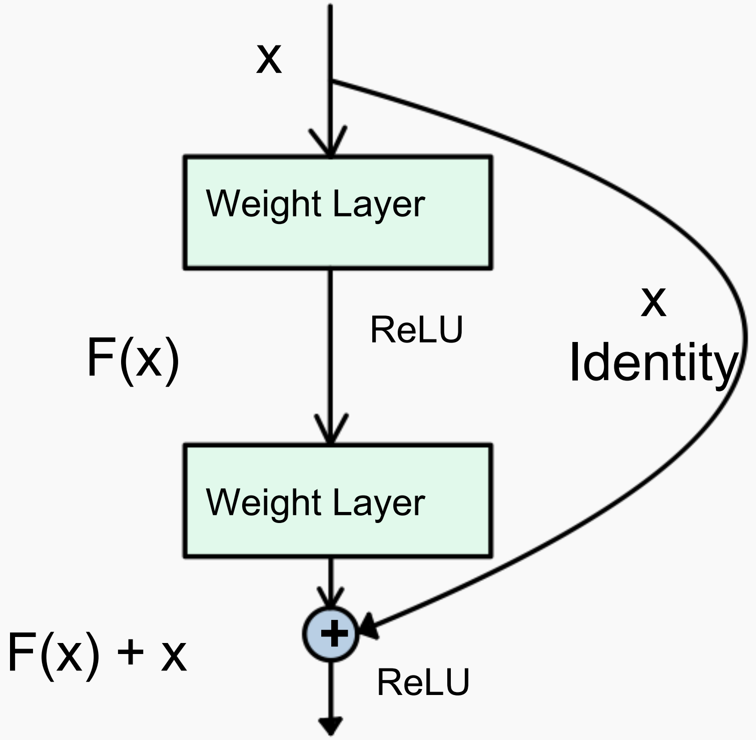


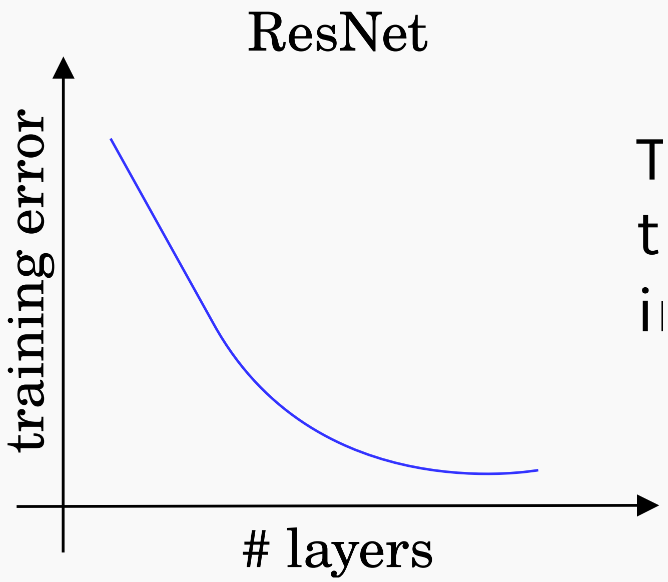
Mandatory Meme

ResNet

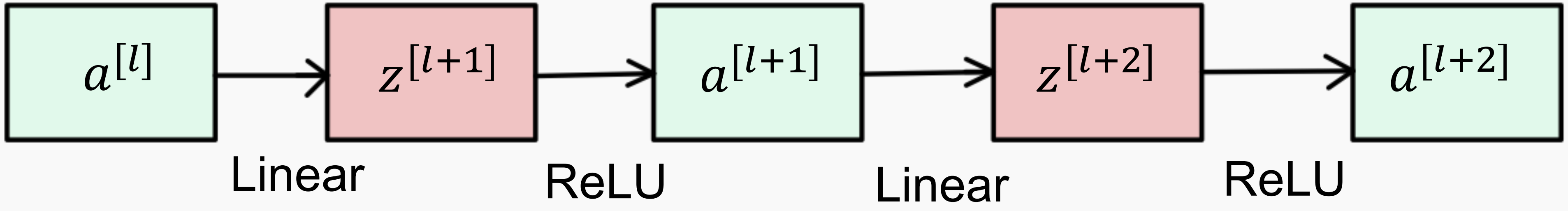
- Presented by [He et al](#) (Microsoft), 2015. Won ILSVRC 2015 in multiple categories. Very similar to Highway Networks [Srivastava et al. 2015](#) introduced the same time.
- Main idea: Residual block. Allows for extremely deep networks.
- Authors believe that it is easier to optimize the residual mapping than the original one. Furthermore, residual block can decide to "shut itself down" if needed.



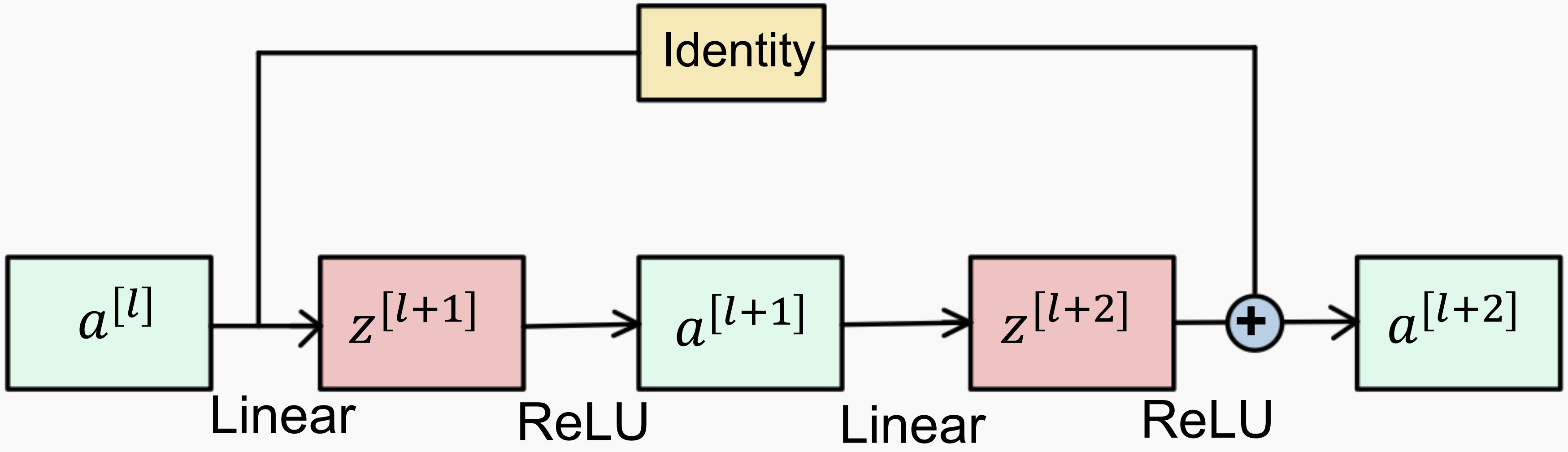


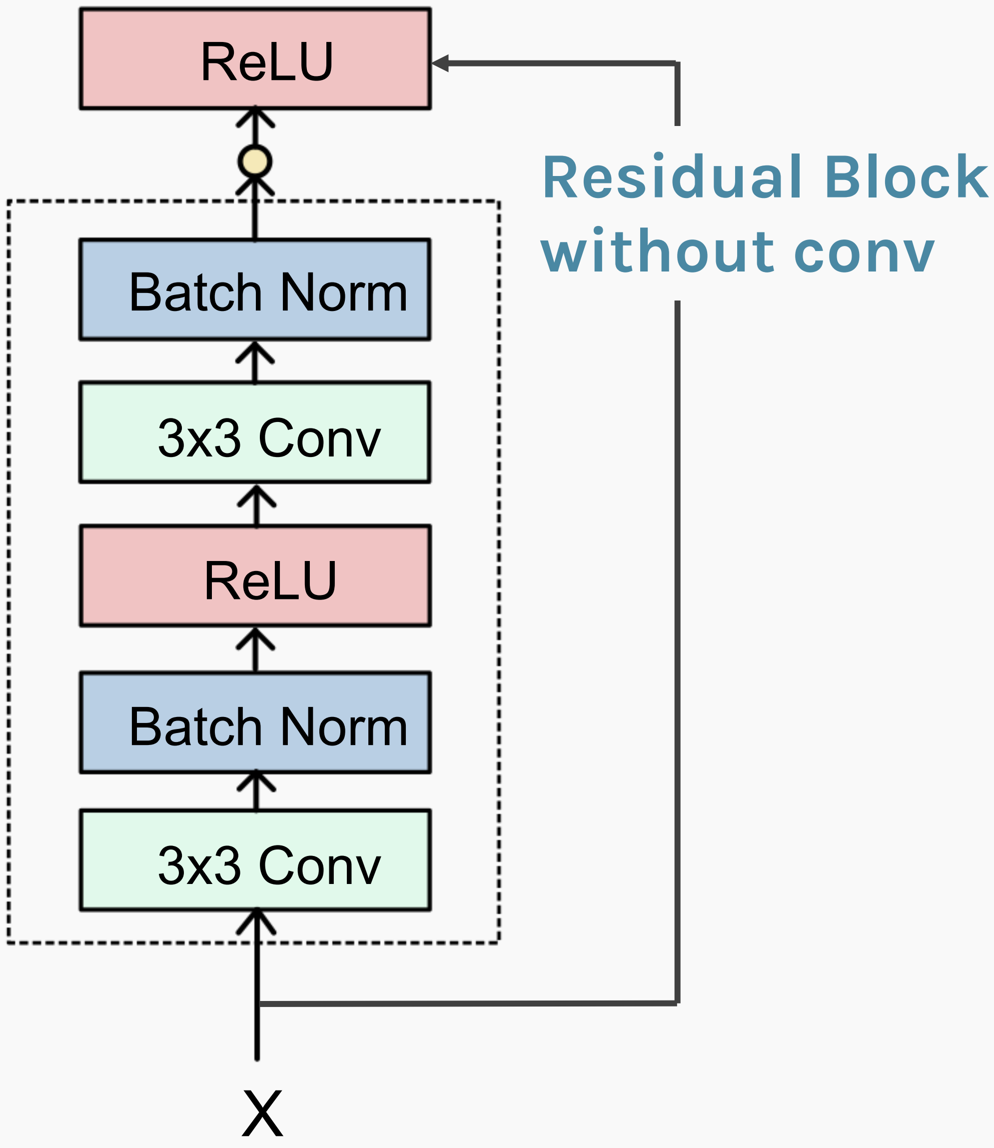


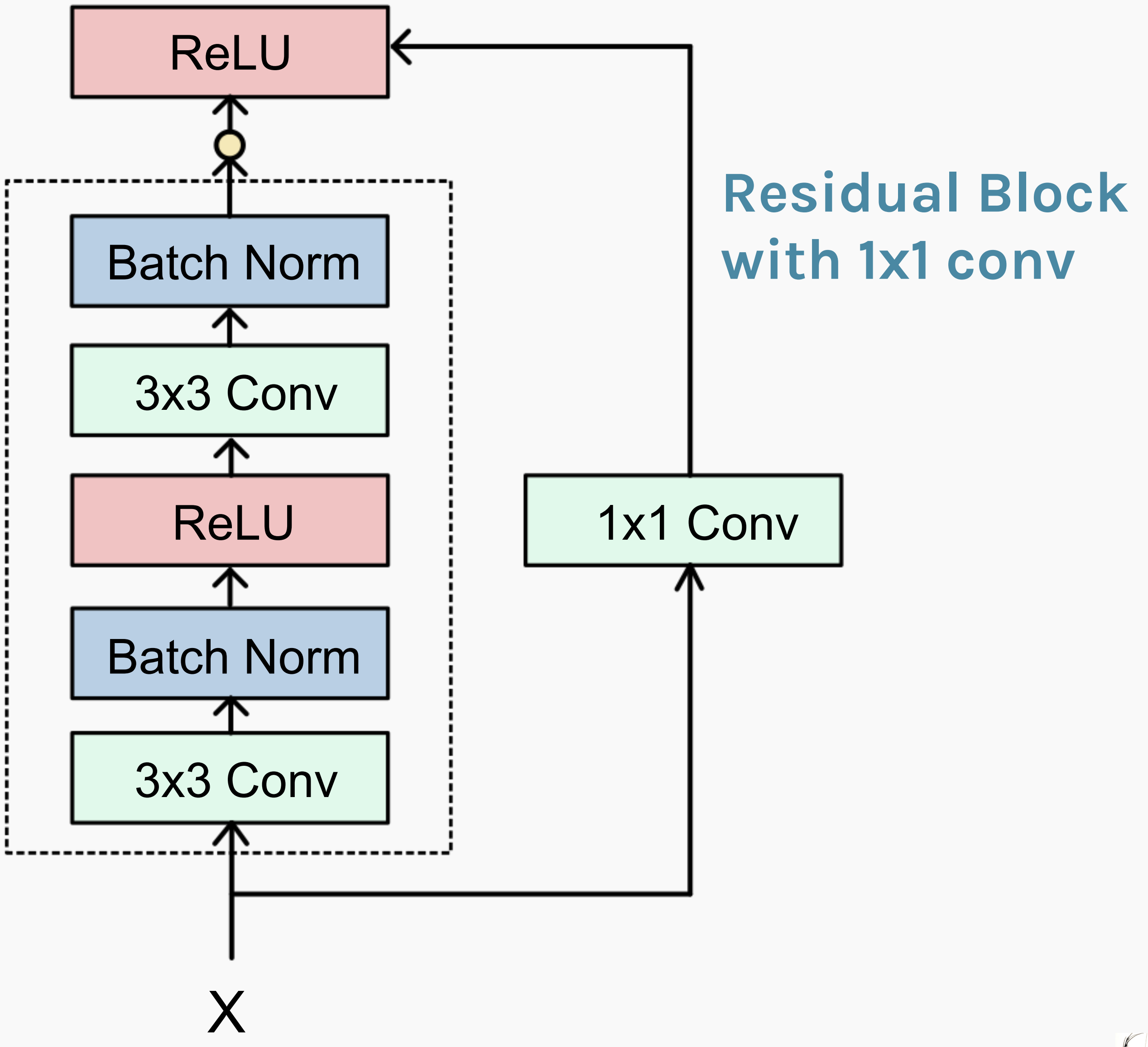
- Residual nets appeared in 2016 to train very deep NN (100 or more layers).
- Their architecture uses ‘residual blocks’.
- Plain network structure:



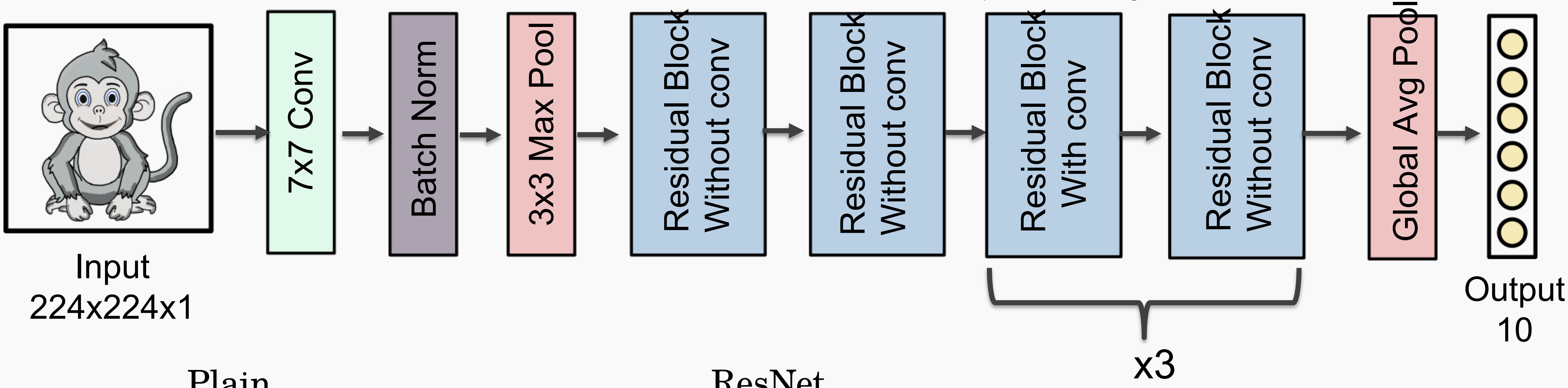
- **Residual network block**

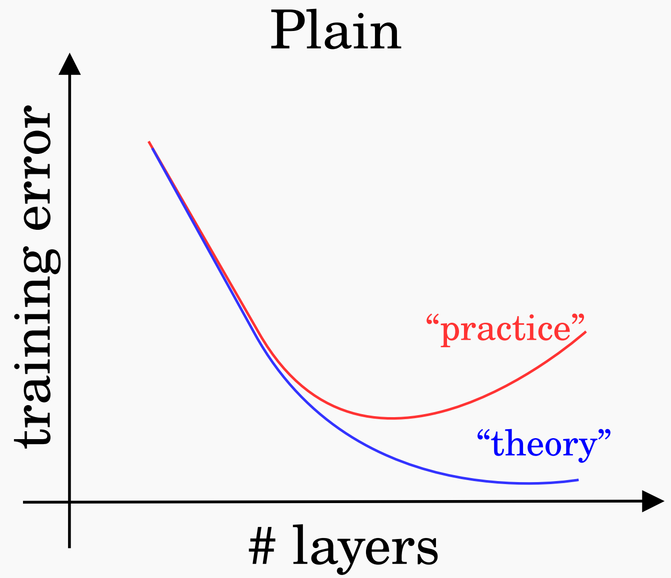


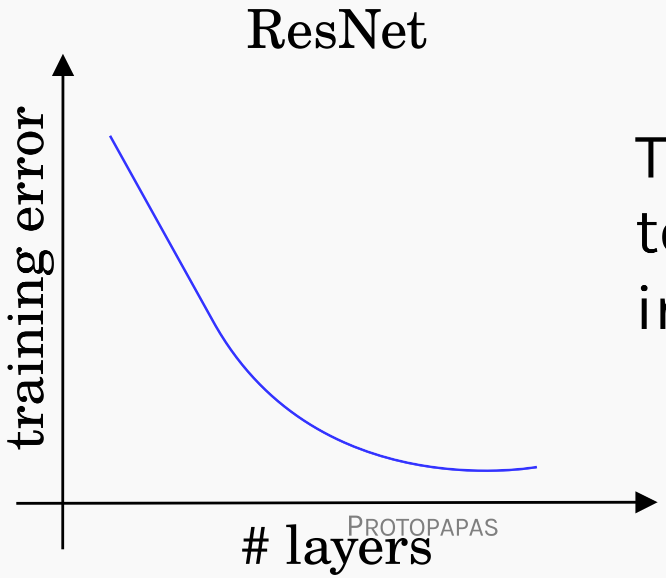

ReLU
Batch Norm
3x3 Conv
ReLU
Batch Norm
3x3 Conv
X
Residual Block
without conv


ReLU
Batch Norm
3x3 Conv
ReLU
Batch Norm
3x3 Conv
1x1 Conv
X
Residual Block
with 1x1 conv

The residual network stacks blocks sequentially





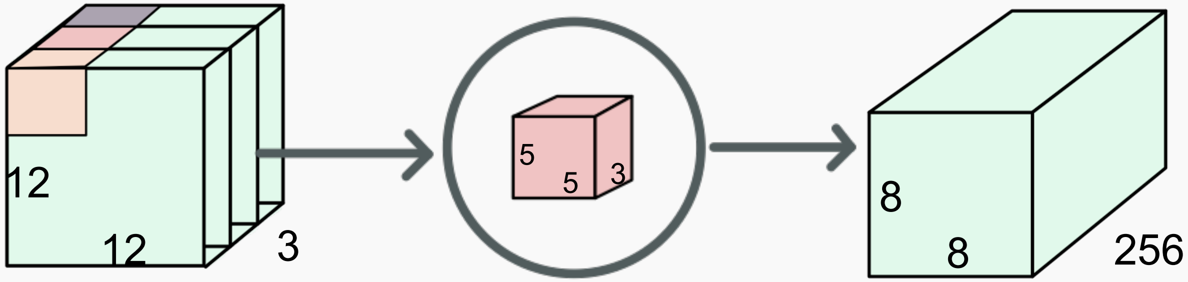


The idea is to allow the network to become deeper without increasing the training time

SOTA Deep Models: MobileNet

Standard Convolution

Filters and combines inputs into a new set of outputs in one step



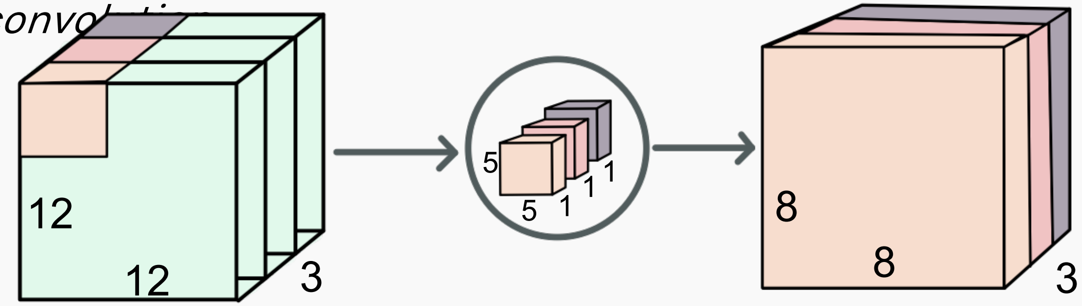
Input: 12x12x3
Filter: 5x5x3x256

Output: 8x8x256
(no padding)

MACs: (5x5)x3x256x(12x12) ~ **2.8M**
Parameters: (5x5x3)x256 + 256 ~ **20K**

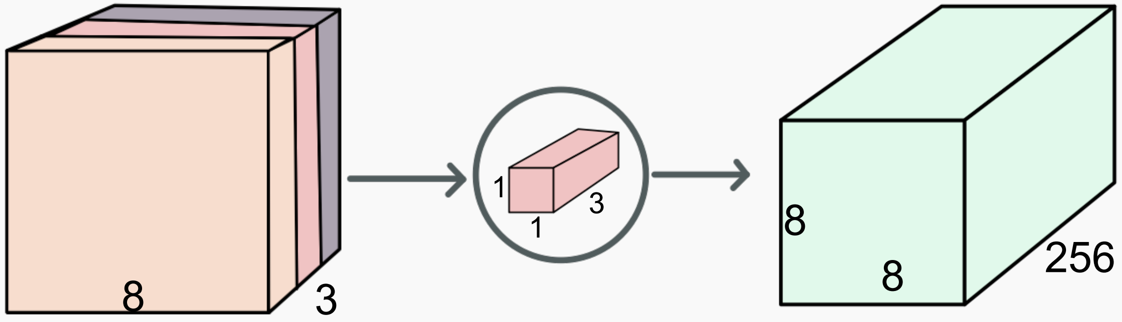
Depth-Wise Separable Convolution (DW)

It combines a depth wise convolution and a pointwise convolution



Input: 12x12x3
Filter: 5x5x3

Output: 8x8x3
(no padding)



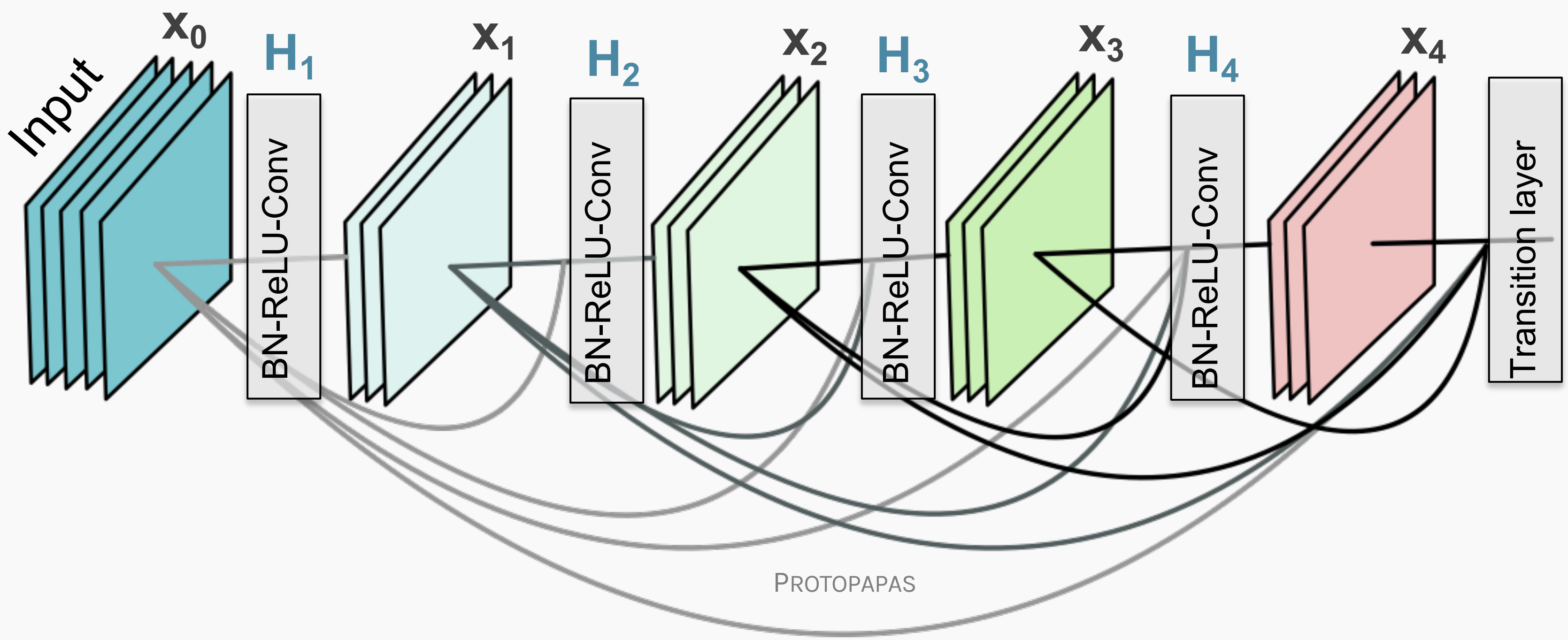
Input: 8x8x3
Filter: 1x1x3x256

Output: 8x8x256
(no padding)

MACs: (5x5)x3x(12x12) + 3x256x(8x8) ~ **60K**
Parameters: (5x5x3 + 3) + (1x1x3x256+256) ~ **1K**

- **Goal**: allow maximum information (and gradient) flow → connect every layer directly with each other.
- DenseNets exploit the potential of the network through feature reuse → no need to learn redundant feature maps.
- DenseNets layers are very narrow (e.g. 12 filters), and they just add a small set of new feature-maps.

DENSE BLOCK

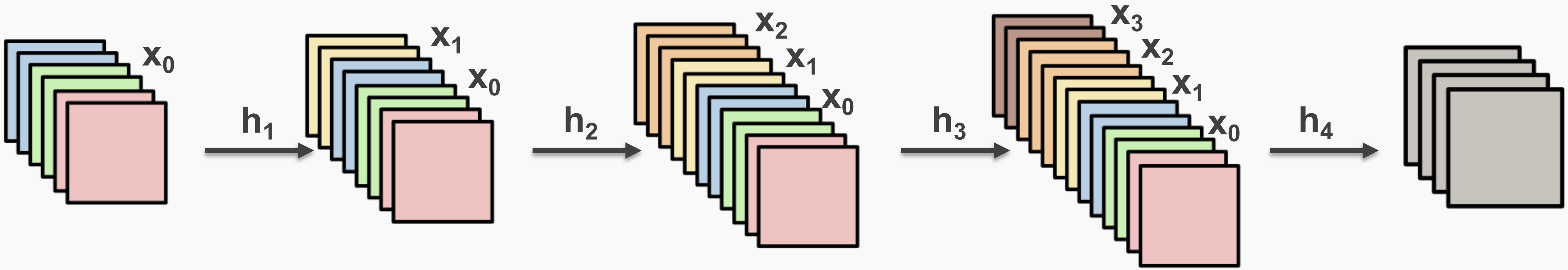


- DenseNets do not sum the output feature maps of the layer with the incoming feature maps but concatenate them:

$$a^{[l]} = g([a^{[0]}, a^{[1]}, \dots, a^{[l-1]}])$$

- Dimensions of the feature maps remains constant within a block, but the number of filters changes between them → **growth rate**:

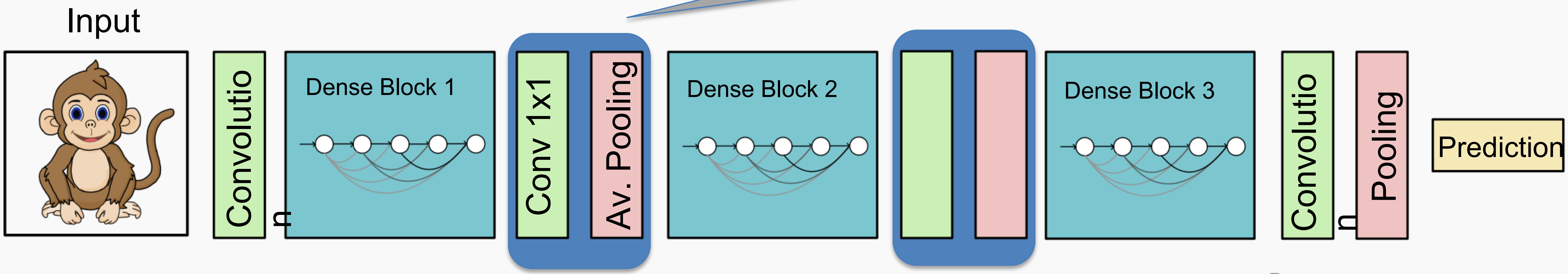
$$k^{[l]} = k^{[0]} + k(l-1)$$

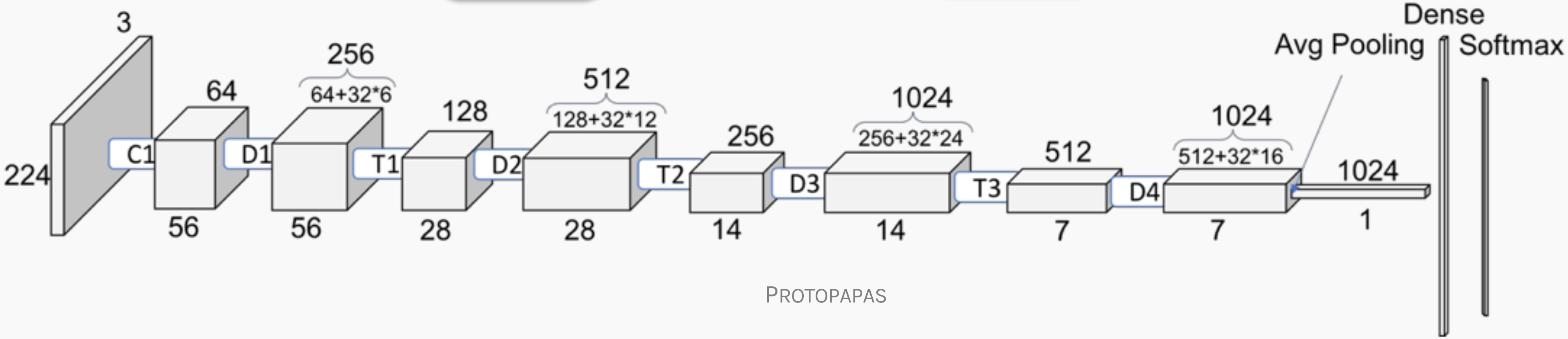


Concatenation during forward propagation

1.Batch Normalization
2.ReLU activation
3.3x3 Convolution

A transition layer is made of:
1. Batch Normalization
2. 1x1 Convolution
3. Average pooling





Beyond

- MobileNetV2 (https://arxiv.org/abs/1801.04381)
- Inception-Resnet, v1 and v2 (https://arxiv.org/abs/1602.07261)
- Wide-Resnet (https://arxiv.org/abs/1605.07146)
- Xception (https://arxiv.org/abs/1610.02357)
- ResNeXt (https://arxiv.org/pdf/1611.05431)
- ShuffleNet, v1 and v2 (https://arxiv.org/abs/1707.01083)
- Squeeze and Excitation Nets (https://arxiv.org/abs/1709.01507)