ANCHORMEN
data activators

COMPUTER VISION

Raoul Grouls

5-JAN-21

TABLE OF CONTENT

COMPUTER VISION TASKS

Image Classification

Classify image

Semantic Segmentation

Classify every pixel into a category

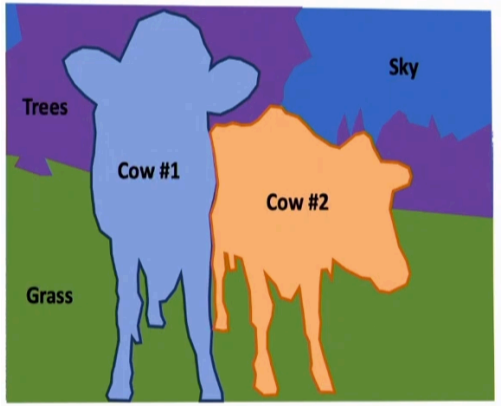
Object Detection

Classify multiple objects on an image and localize them

Instance Segmentation

Classify multiple objects on an image, localize them and make pixel-wise object mask

Panoptic Segmentation



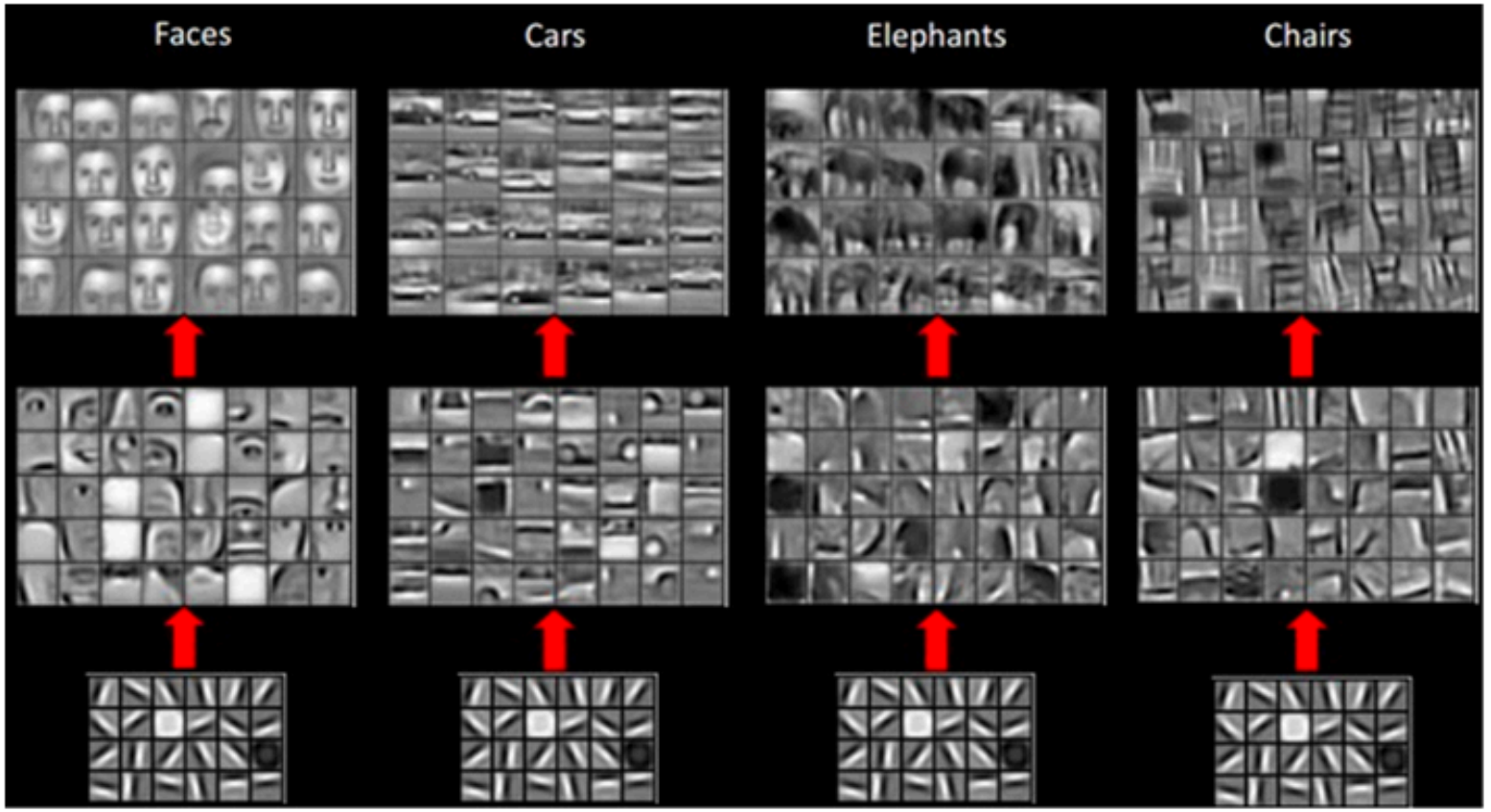
Instance segmentation for “things” and semantic segmentation for “stuff”

MOTIVATION FOR CONVOLUTIONAL LAYER

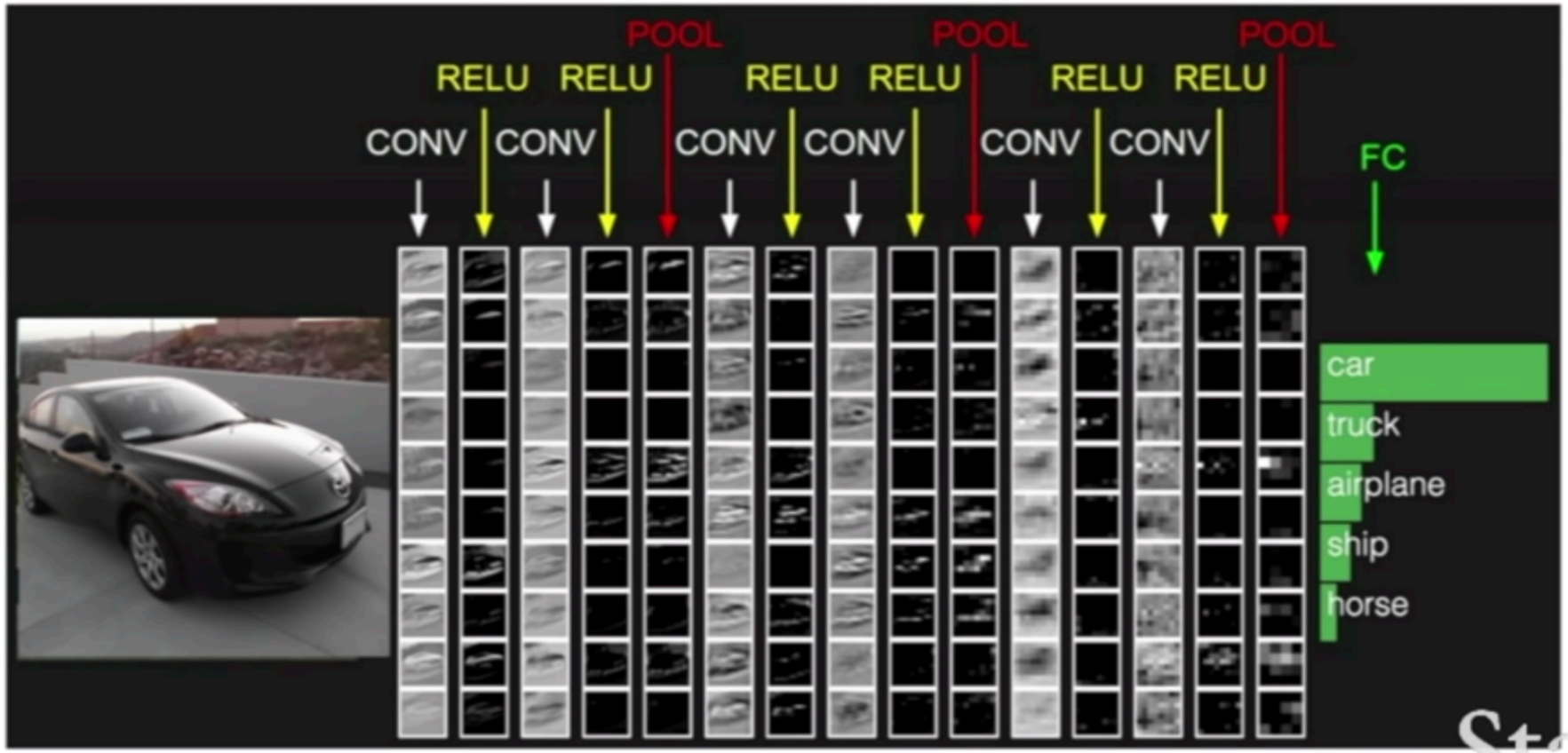
- Preserve spatial structure
  - We have been flattening the image (eg 28x28) into one vector (eg 784). With this, we destroy a lot of spatial information.
  - Idea: Don’t flatten the image to one vector, but process the ‘context’ of a pixel.
  - Nearby pixels will stay nearby
- Translational invariance
  - If you shift the position of an object, it still means the same thing
  - A cat in the right corner of the image is also a cat in the left corner
  - So we want to share parameters

MORE AND MORE COMPLEX FEATURES

- First layers more simple features
- Latter layers more complex features



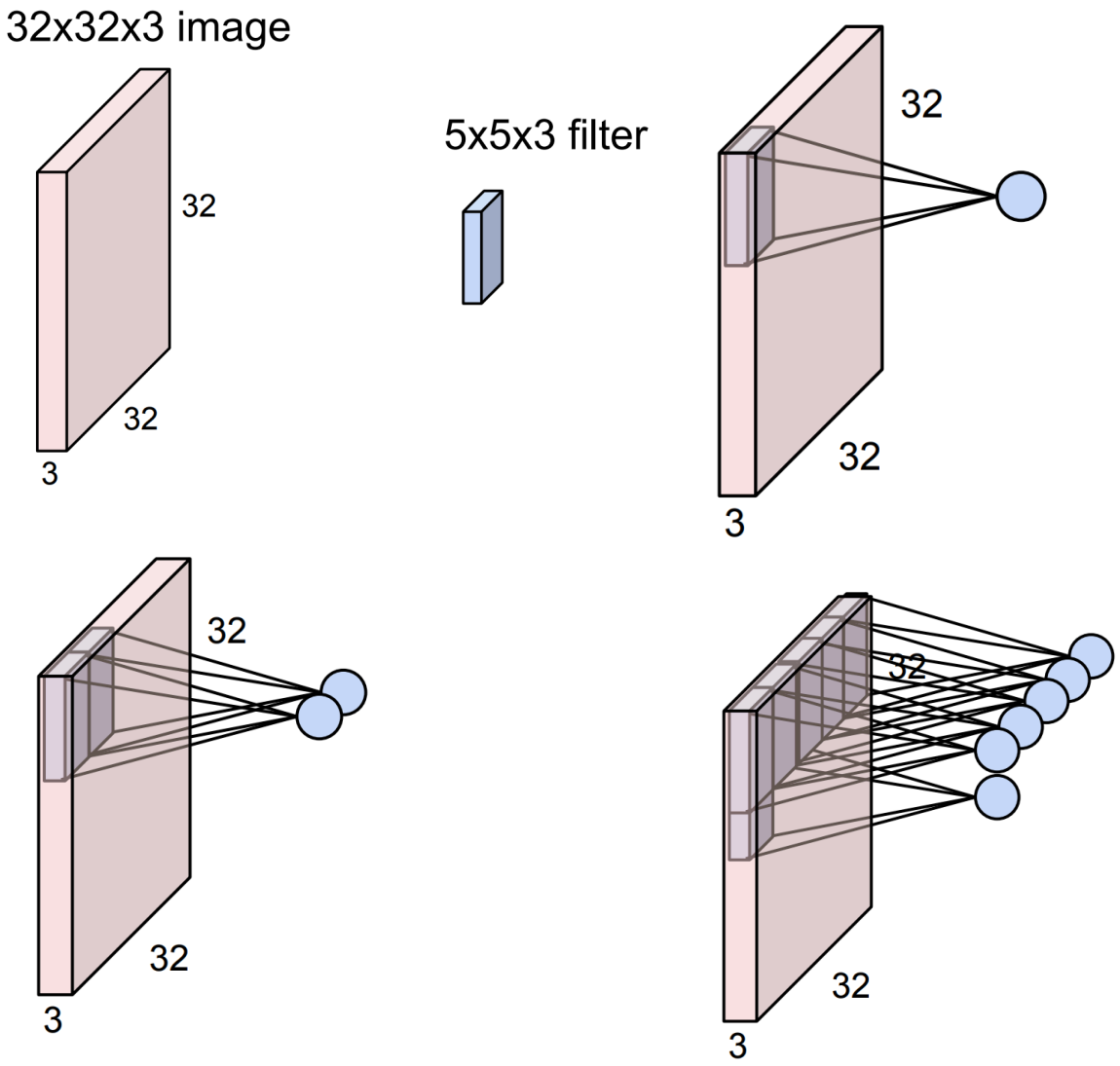
BASIC CONVOLUTIONAL NETWORK STRUCTURE



CONVOLUTIONAL NET COMPONENTS

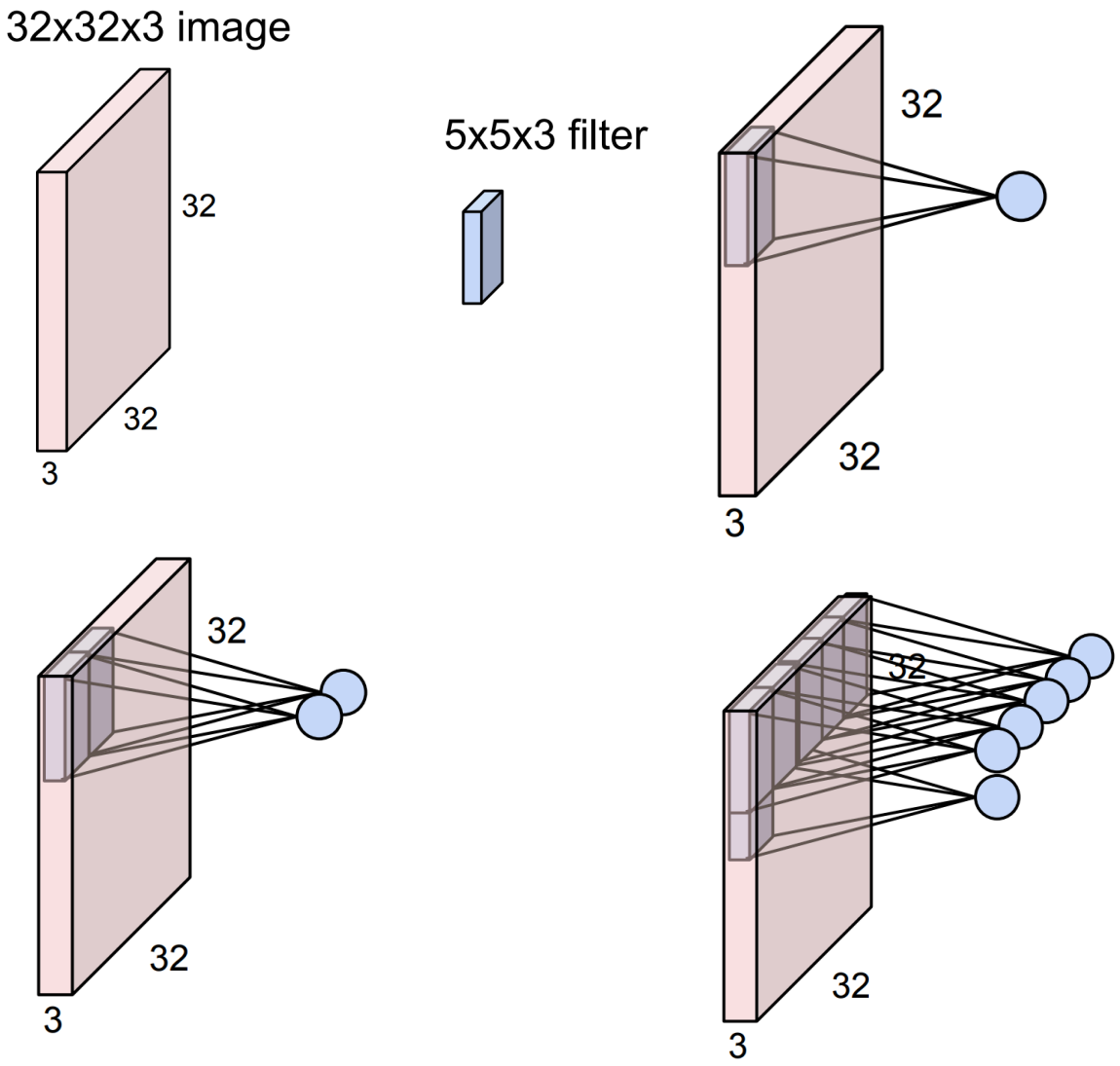
CONVOLUTIONAL LAYER

- Take a filter
- Slide over every position of the input



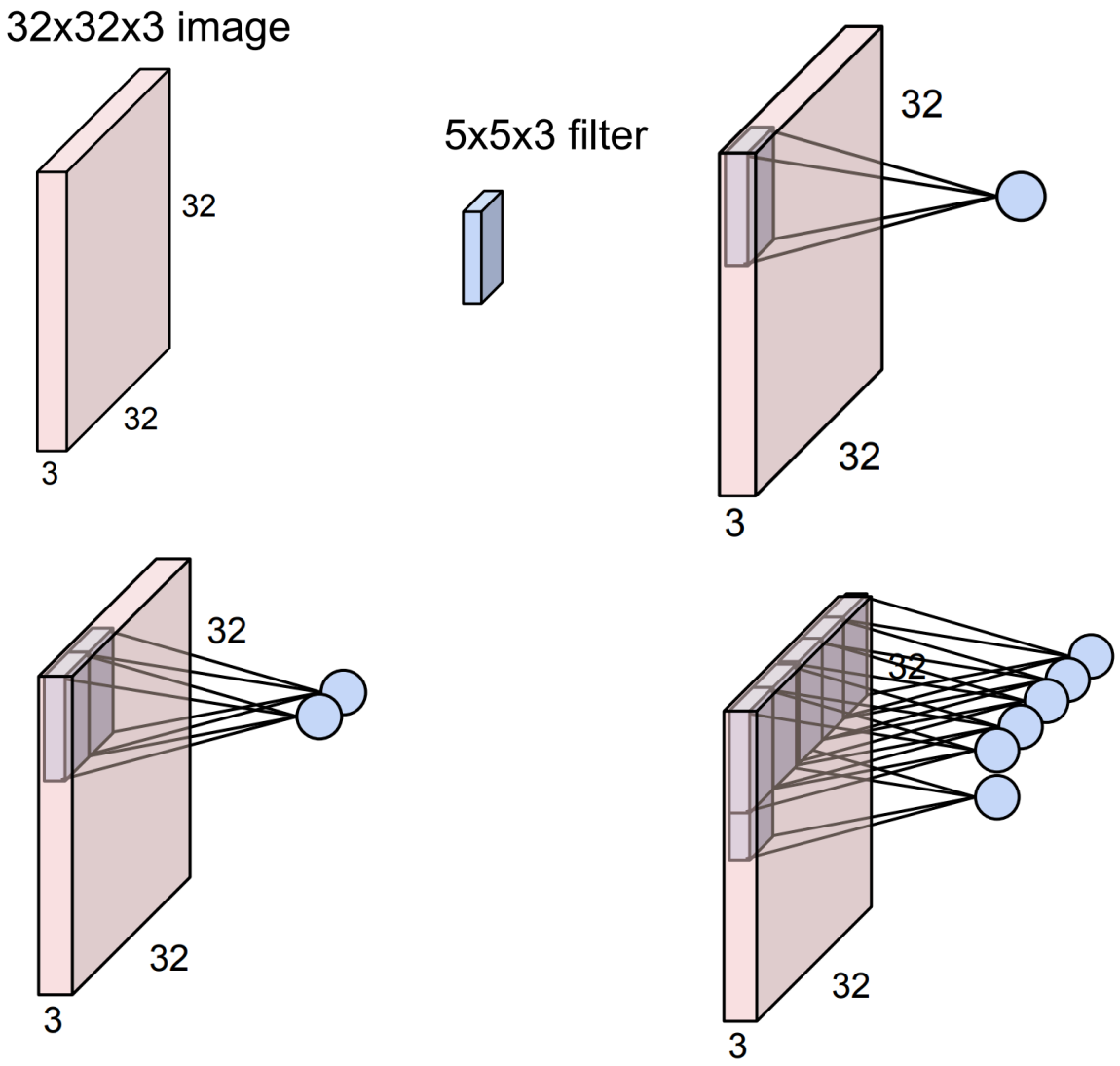
CONVOLUTIONAL LAYER

- Calculate the dot product between the filter and input segment the filter covers
- Arrange result of the dot products in a 2D array, yielding the output activation map
  - The activations in the activation map should reflect the positions of the filter, when the dot product was calculated



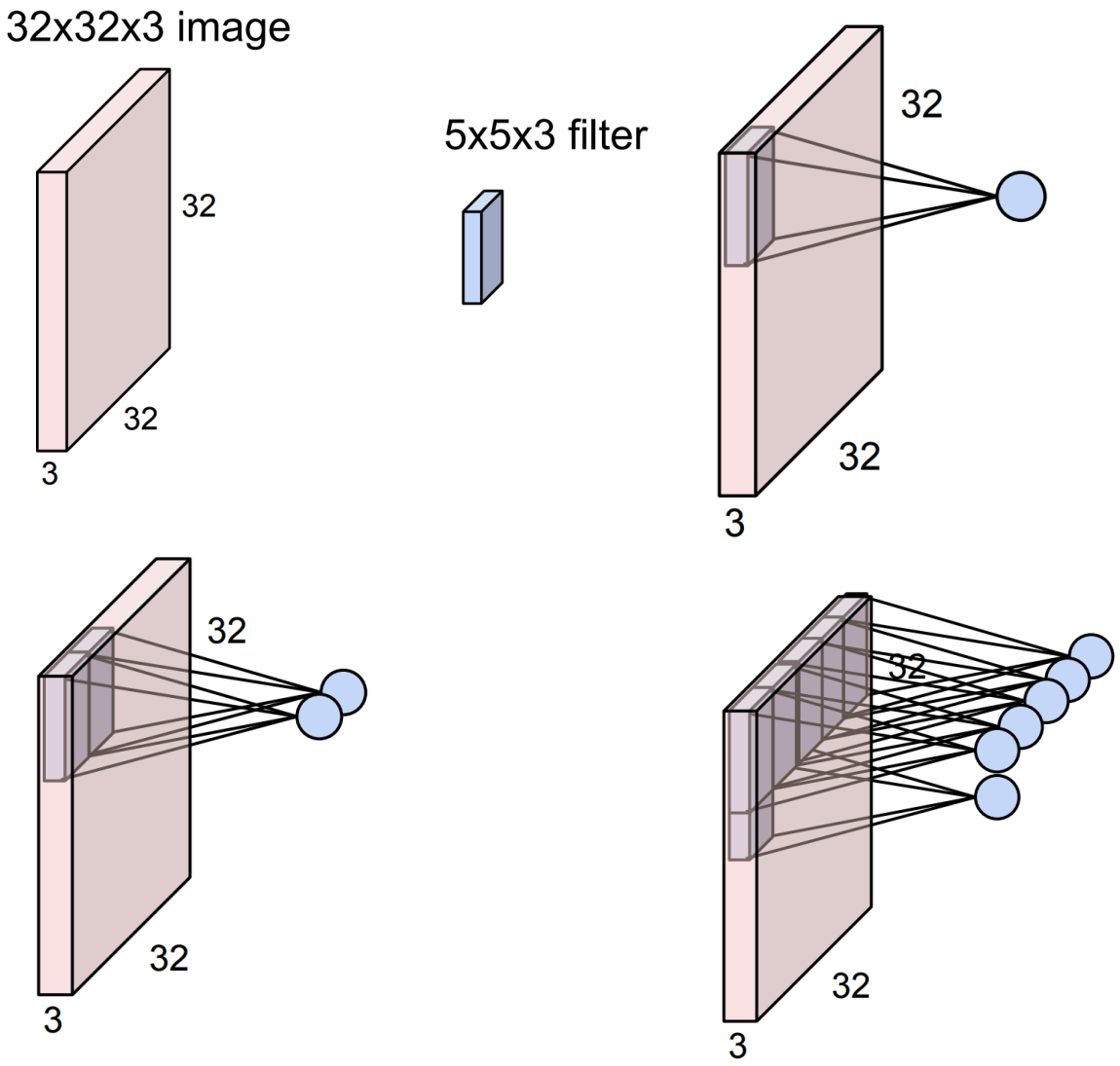
CONVOLUTIONAL LAYER

- The filter goes through the entire depth of the input
- Parameters in filter: (width, height, channel number)
- In case of first hidden layer, the channel number is 3, representing the RGB colours

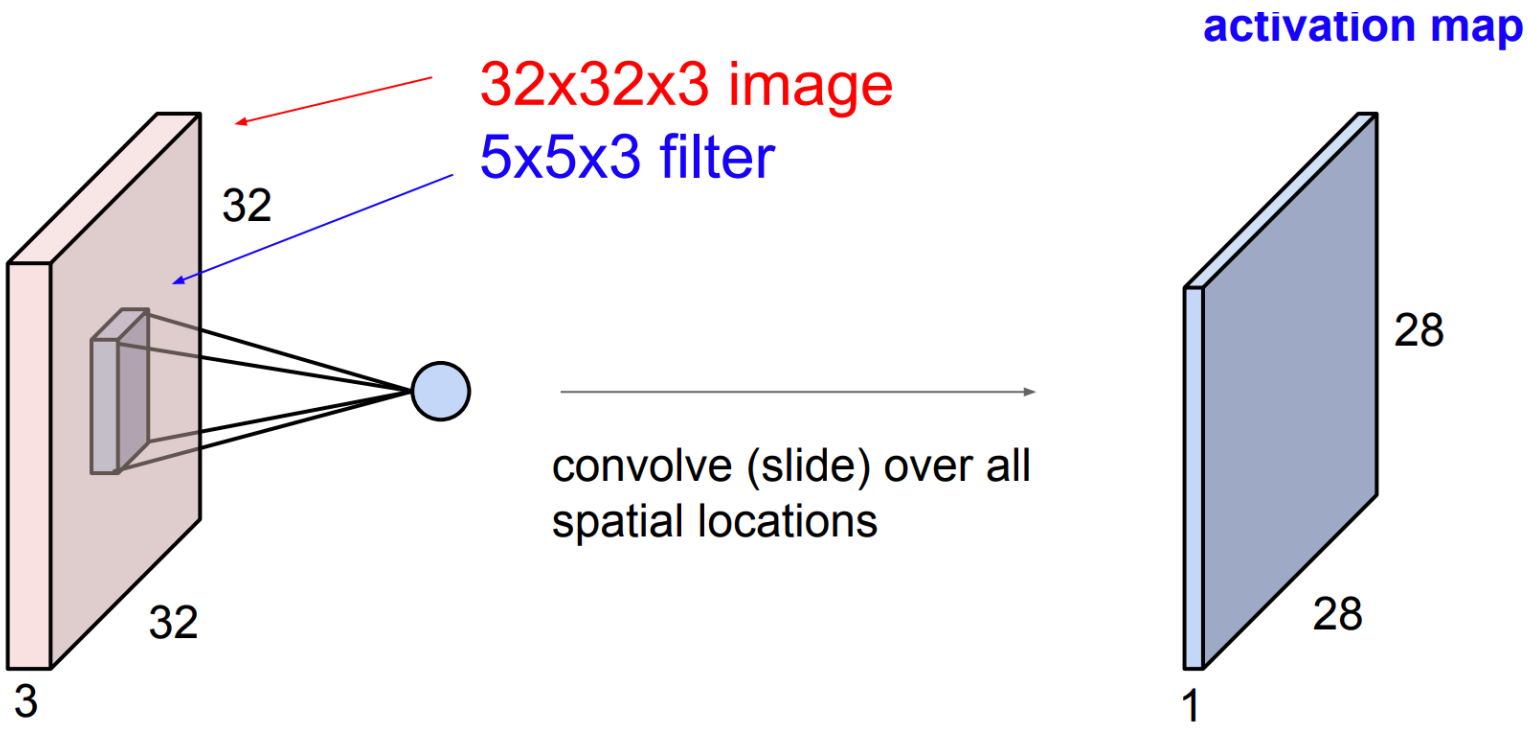


CONVOLUTIONAL LAYER

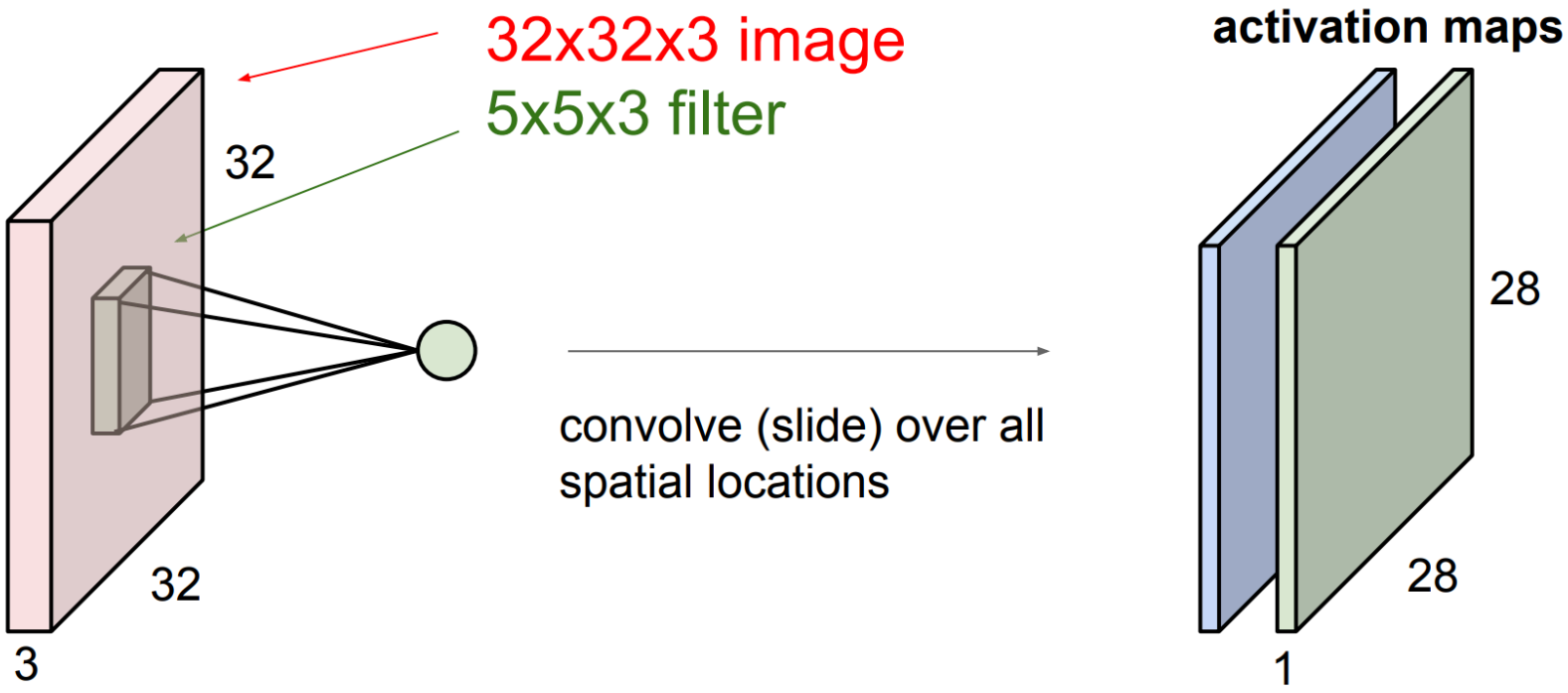
- Later on, the channel number represents the amount of filters that produced feature maps.
- So, we could end up with small feature maps (eg 5x5 pixels big) but we might have 512 of those.



CONVOLUTIONAL LAYER

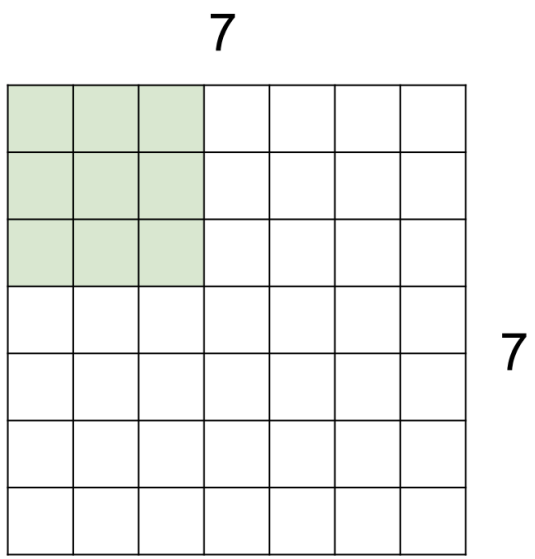


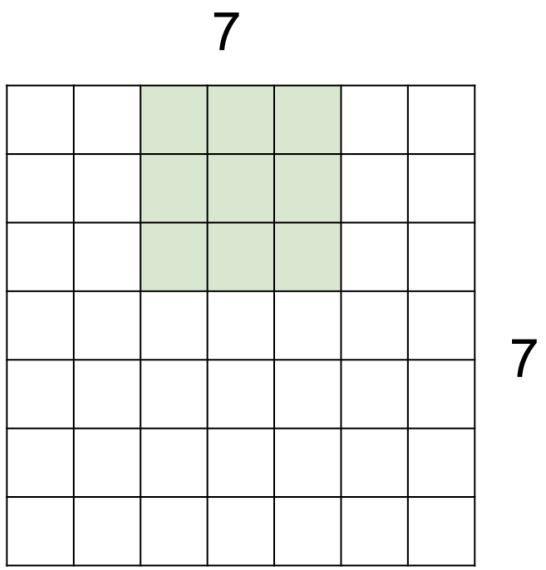
CONVOLUTIONAL LAYER

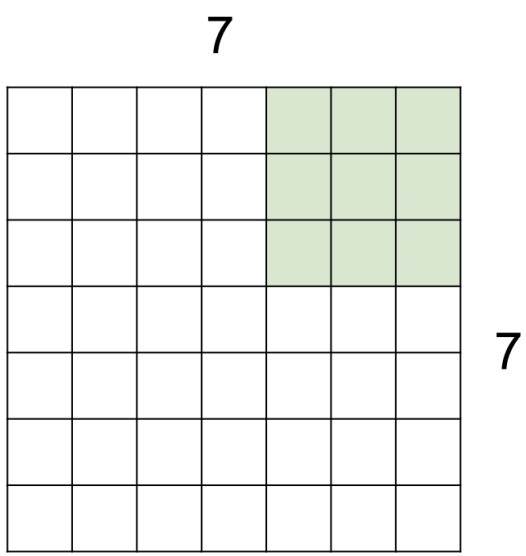


STRIDE

- Stride: how many pixels to step when sliding the filter
- Shrinks the output dimensionality, if stride > 1
  - If stride is 2, dimensionality is approx. halved

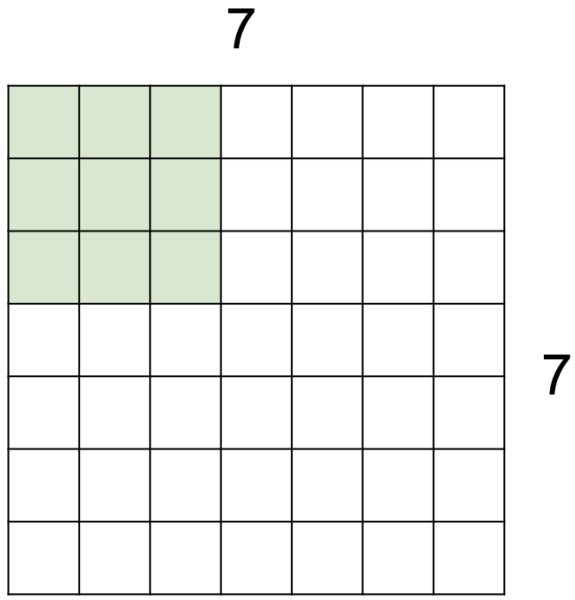


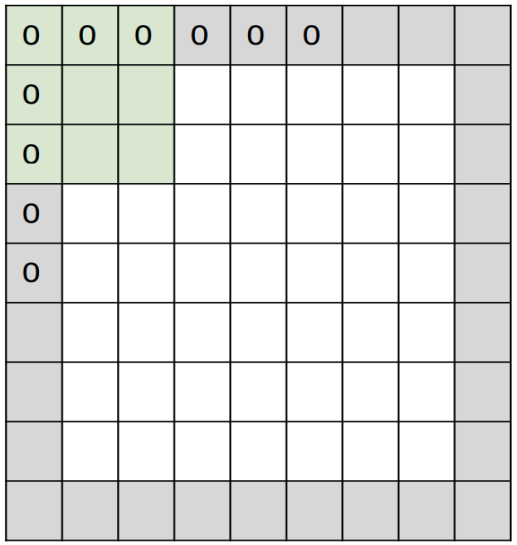




PADDING

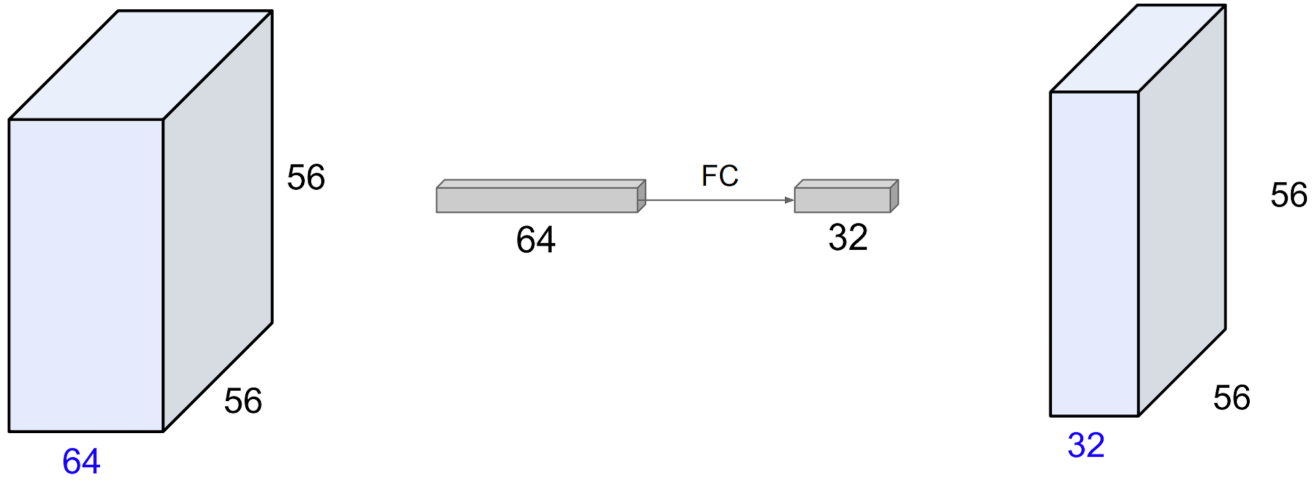
- No padding
  - Then slide as far as we can, ignore the last pixels if one more filter does not fit
  - Output size will be smaller than input size
- Padding
  - Then filter can slide out of the image
  - Output size will be the same as input size
  - Why: because otherwise for a deep network the activation layer would shrink very quickly





1X1 CONVOLUTIONAL LAYER

- A fully connected layer in between channels
  - Weights are shared in between all spatial positions
- Preserves spatial dimensions
- Can be used to reduce the number of channels
  - “Bottleneck layer”

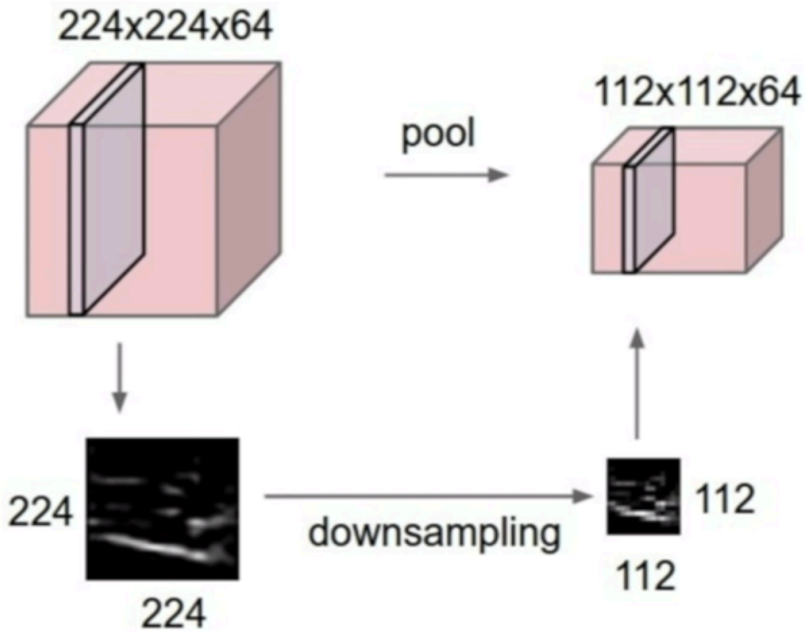


CONVOLUTIONAL LAYER DESIGN CHOICES

- Number of filters
- Filter spatial size
- Stride
- Padding

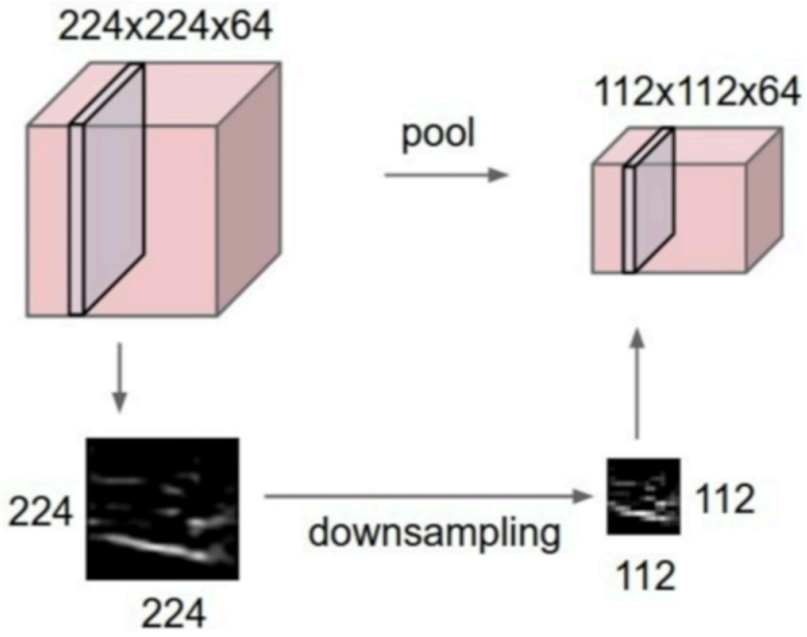
POOLING

- To downsample the input
- Swipes through the input and performs the aggregating operation
- Does this in every channel independently
  - Doesn’t effect depth dimensionality, unlike the convolutional layer



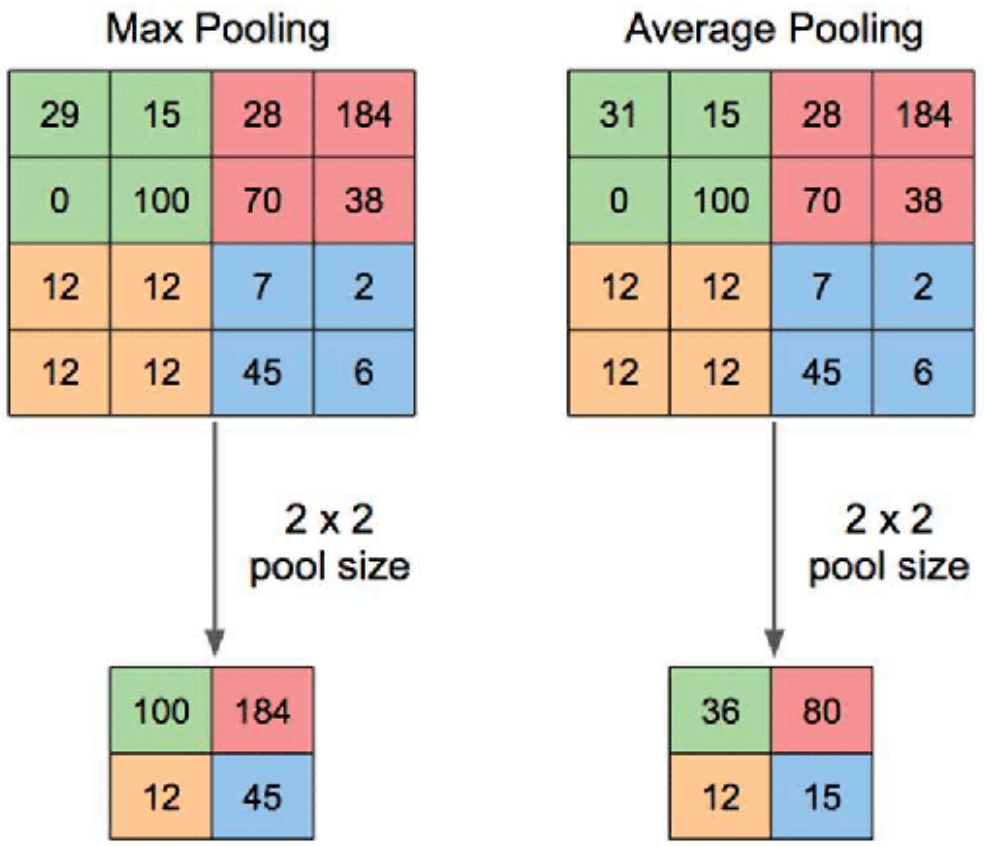
POOLING

- Pooling, or convolution with stride 2?
  - Contains no learnable parameters
  - Introduces some translational invariance
  - But sometimes conv with stride > 1 works better, and some architecturs remove maxpooling altogether in favor of stride=2



POOLING TYPES

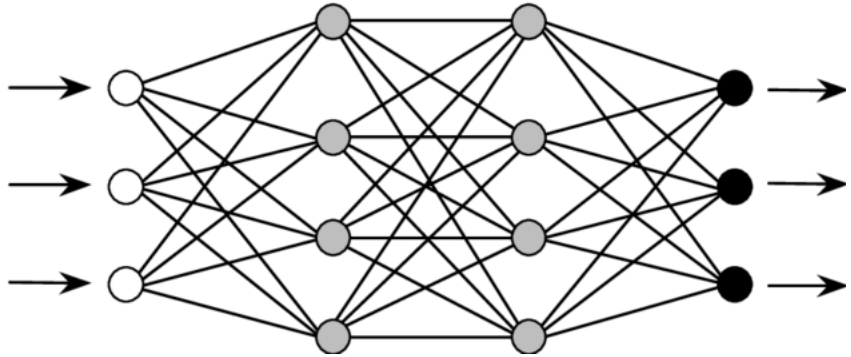
- Max
  - Intuition: If there was any high feature value in the area, we only care about that, not the other values
  - More translational invariance
- Average
  - Intuition: Makes a more smooth downsampling
  - Keeps the spatial structure more
- Usually max is preferred over average pooling



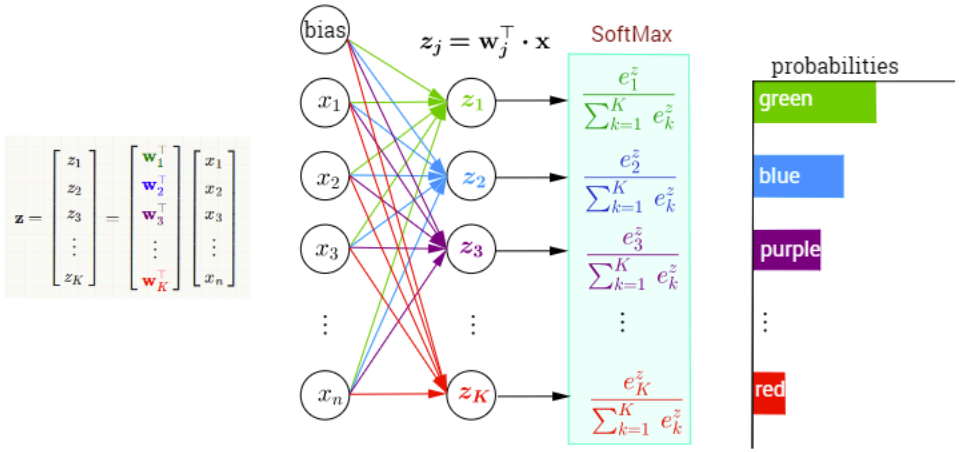
POOLING DESIGN CHOICES

- Filter size
- Stride
- Type
  - Max
  - Average
- Common settings
  - Max, F=2, S=2
  - Max, F=3, S=2

FULLY CONNECTED LAYERS, SOFTMAX



- Flatten the last activation maps into a single 1D vector
- The output layers should either use a softmax for classification, or use no activation at all.
- Softmax turns the logits (ranging from –inf to +inf) into a probability distribution (positive numbers that add up to 1 over all the output neurons)
- If you don't use a softmax, you should set `from_logits=True` in your loss function.



$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

TRAINING

TRANSFER LEARNING

- Sometimes you overfit, because you don’t have enough data
- Idea: Pretrain model on a large generic dataset, then refine on your problem specific data set
- Motivation: the features that the model learned from pretraining will be generic, and useful for your data set

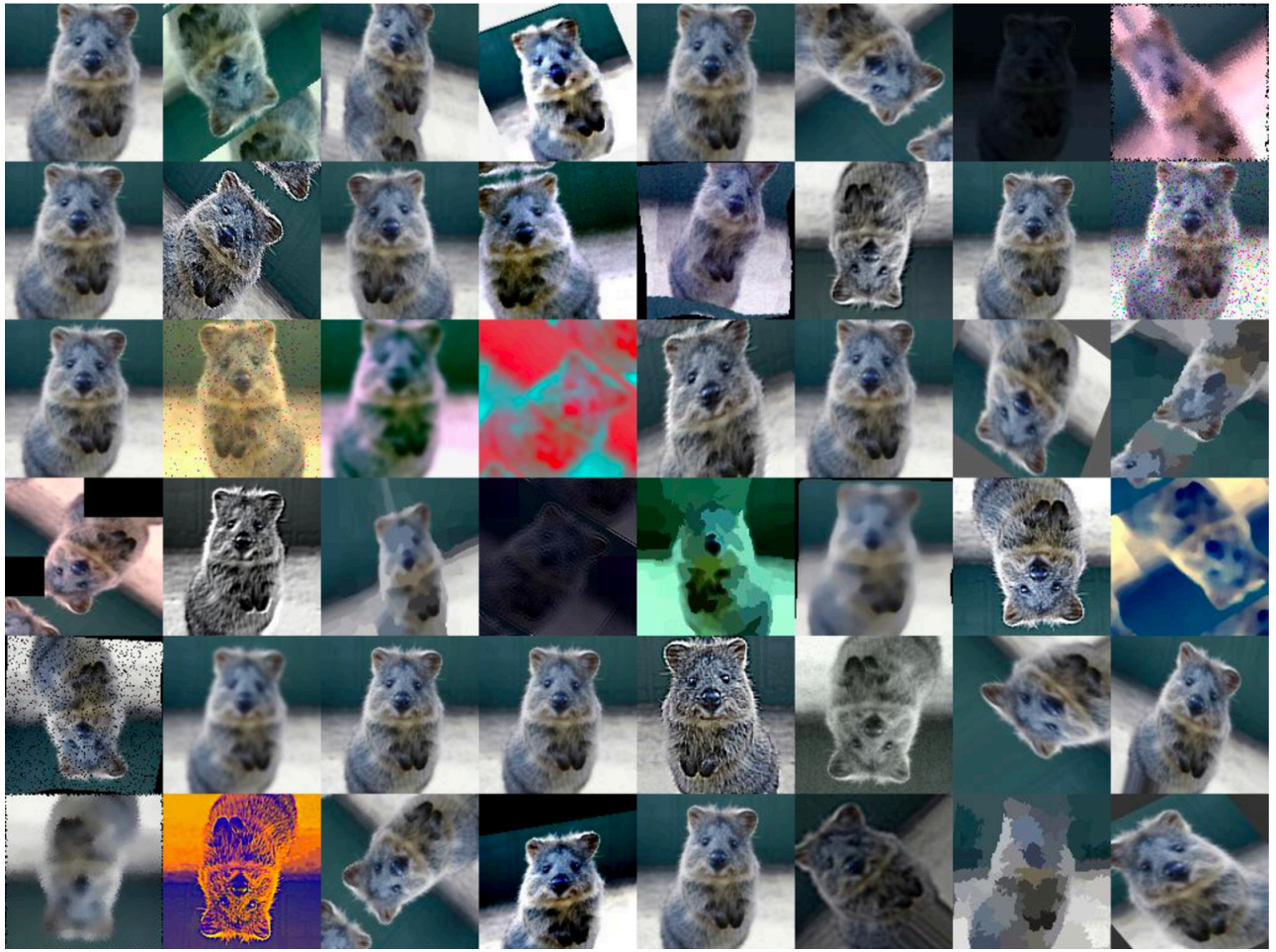
TRANSFER LEARNING

- Strategies
  - Pretrain and freeze
    - Freeze weights, except from last layer(s)
    - Train last layer(s) with your specific dataset
  - Pretrain and refine (can be a second step)
    - Don’t freeze weights, just refine with your data
    - Keep the learning rate low, for refining training

- The less data you have the smaller part of the model to refine (for very few data, only the last layer)
- Transfer learning (pretraining) nowadays is a norm, not an exception

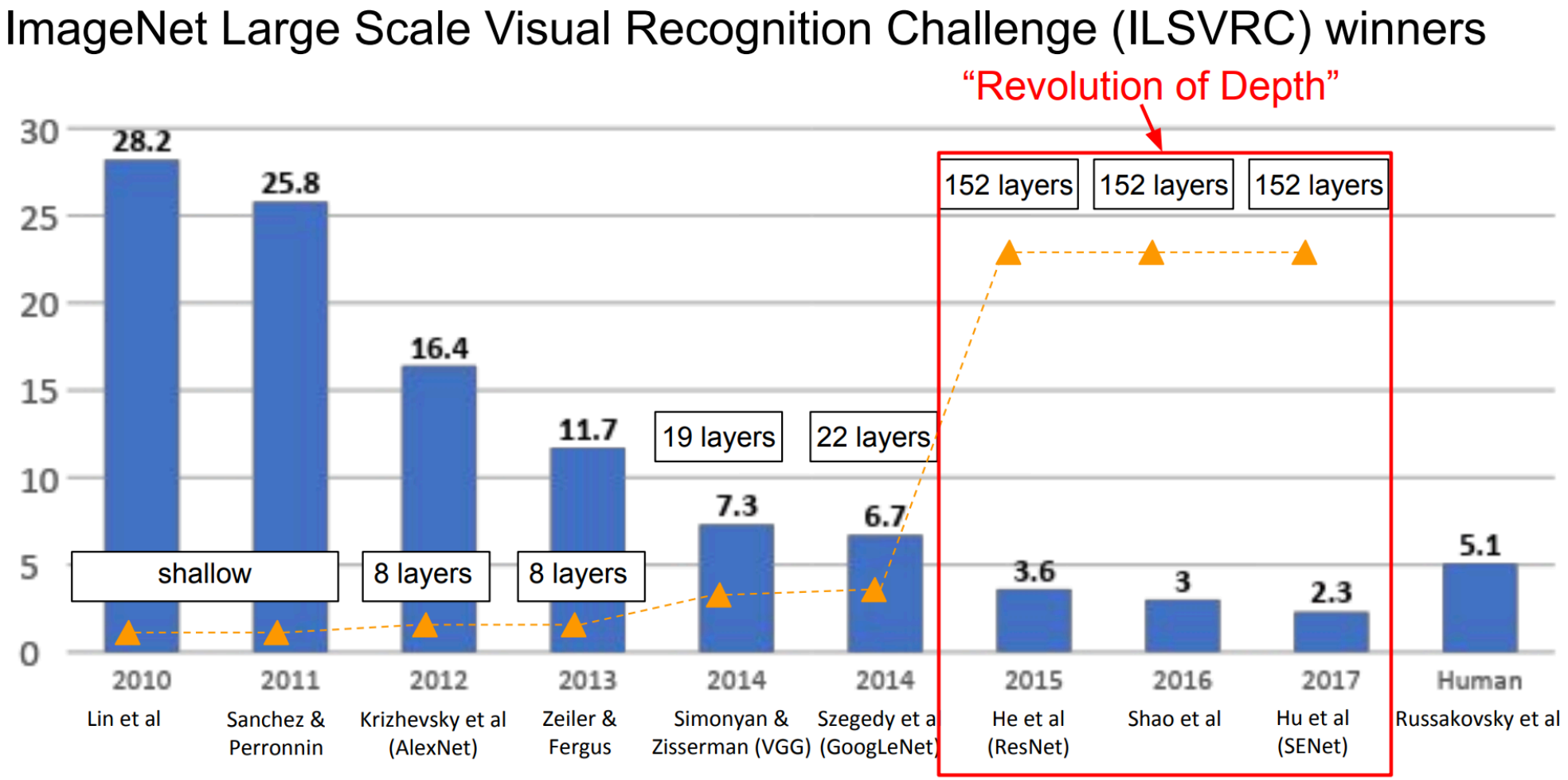
DATA AUGMENTATION

- Idea: randomly transform the image
- Reasoning:
  - Free way of generating new training data
  - Acts as a regularization
    - NN doesn't see the same pixel combination over and over again
- Transformations
  - Flip
  - Scale
  - Crops
  - Change contrast, brightness
  - Colour jittering
  - Salt and paper
  - Cutout (cover parts of the image with black rectangle)



CNN ARCHITECTURES
IMAGE CLASSIFICATION

CNN LAYER NUMBERS



LENET AND ALEXNET

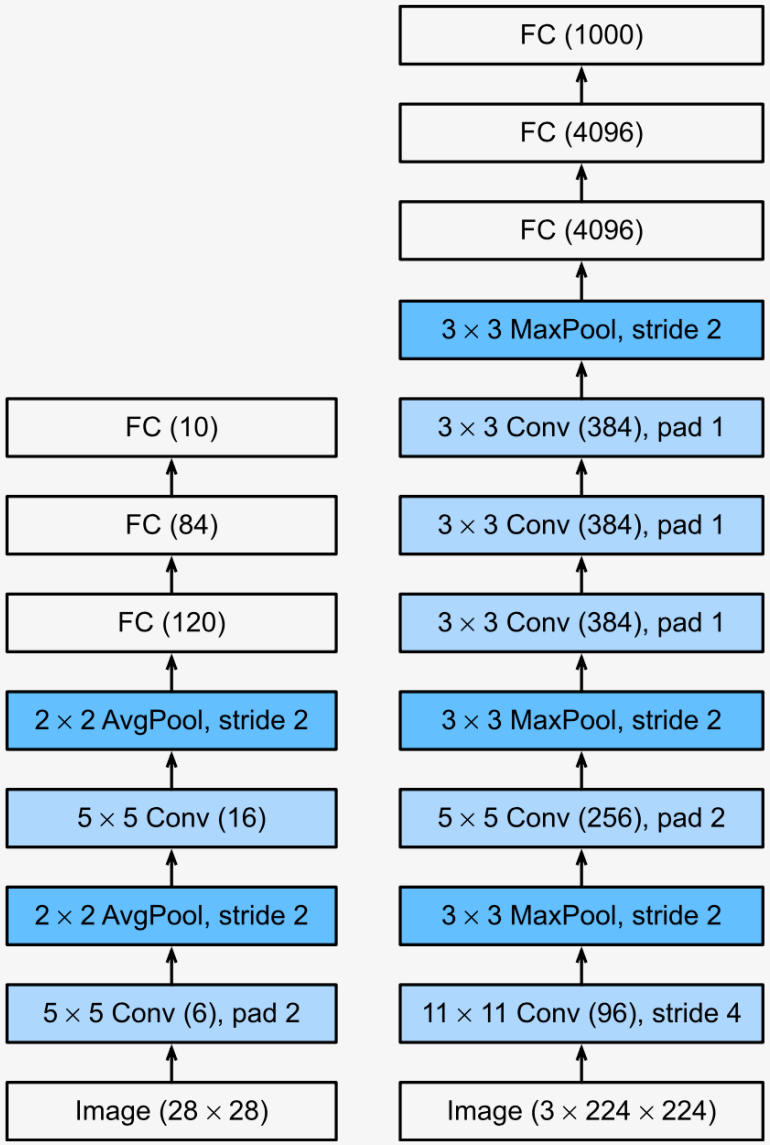
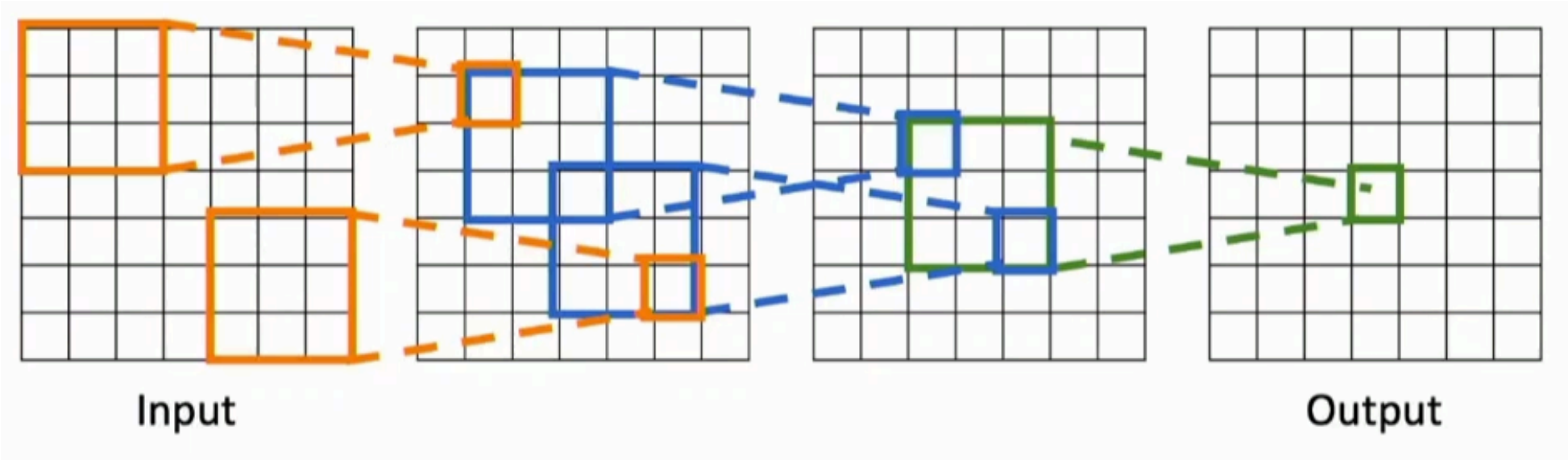


Fig. 7.1.2 From LeNet (left) to AlexNet (right).

EFFECTIVE RECEPTIVE FIELD

- Effective receptive field is the total area of inputs, a set of layers is looking at combined, to calculate one element of the activation map

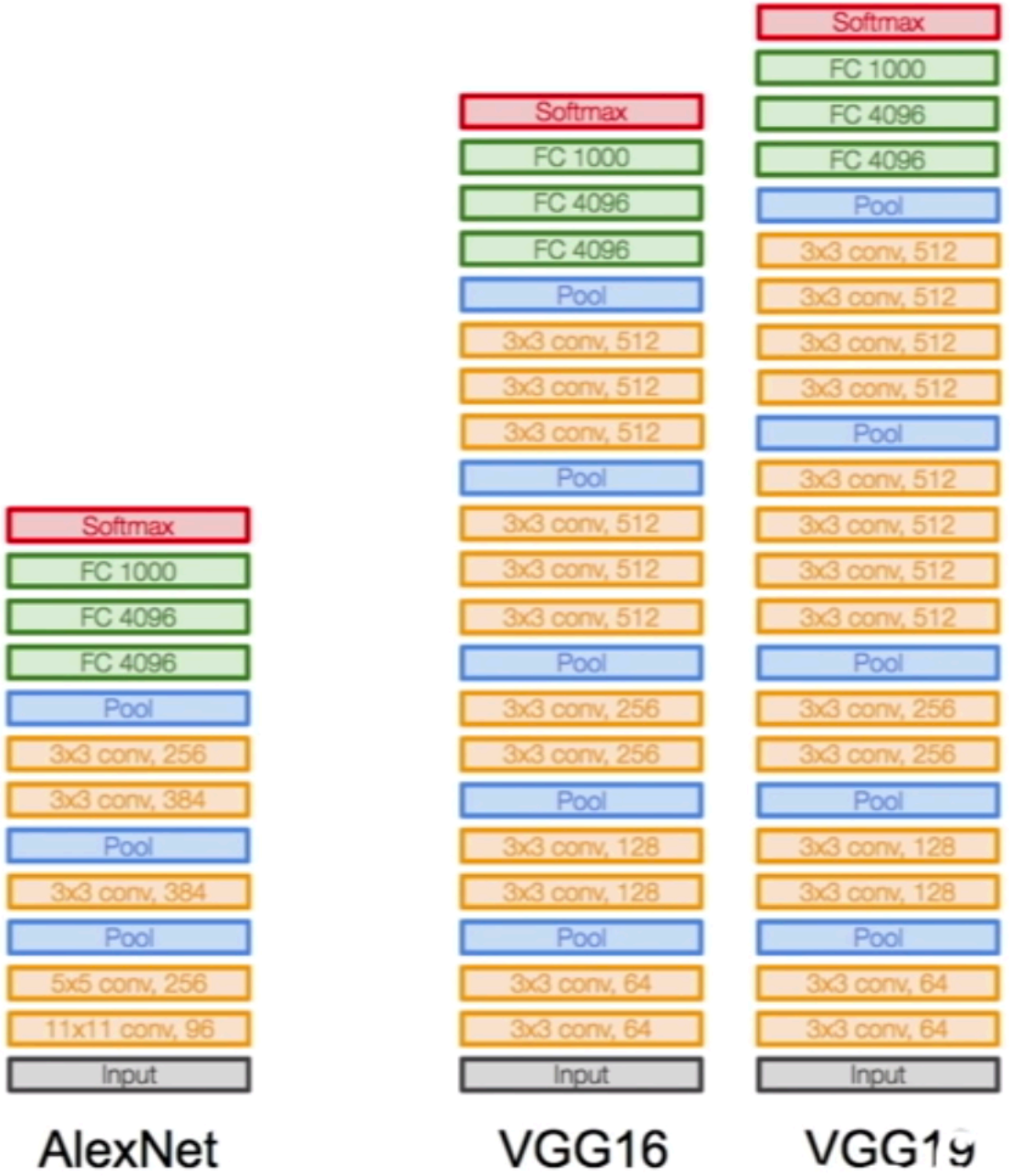


The effective receptive field of 3 stacked 3X3 filters is 7

VGG

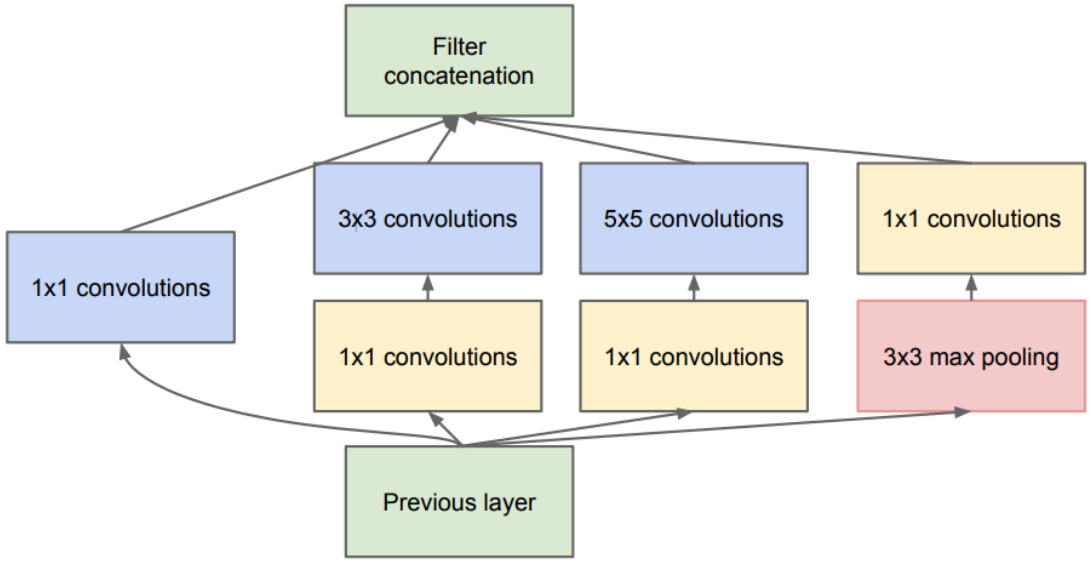
- Design principals
  - Smaller filters, but deeper
  - Double channel number after each pool
  - deeper, more nonlinearities

- Double channel number after each pool
  - Keeps constant compute over depth (pool reduces spatial resolution)



GOOGLENET

- Idea: Inception module
  - Parallel filters
  - Bottleneck layers
    - Can reduce depth arbitrarily
    - Otherwise would require too much computation



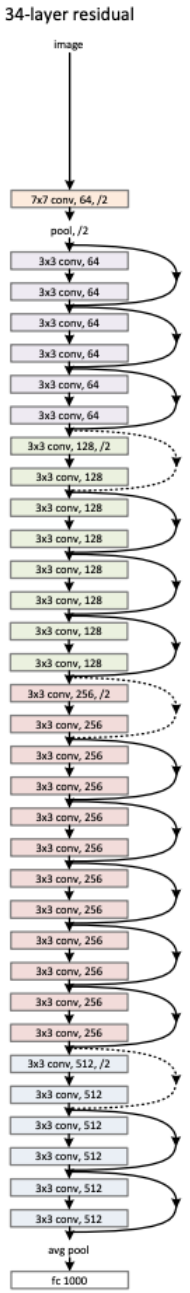
GOOGLENET

- Some properties
  - Focuses on efficiency
  - 22 layers
  - Only 5M parameters
    - Alex net had 16M parameters, but GoogleNet is deeper

RESNET

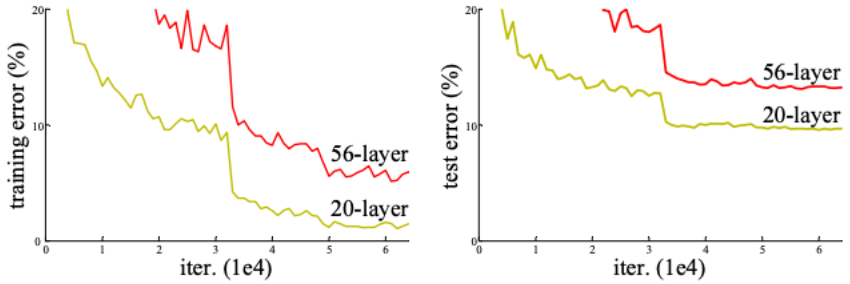
2 important discoveries, made deep networks trainable

- Batch normalization
- Residual block



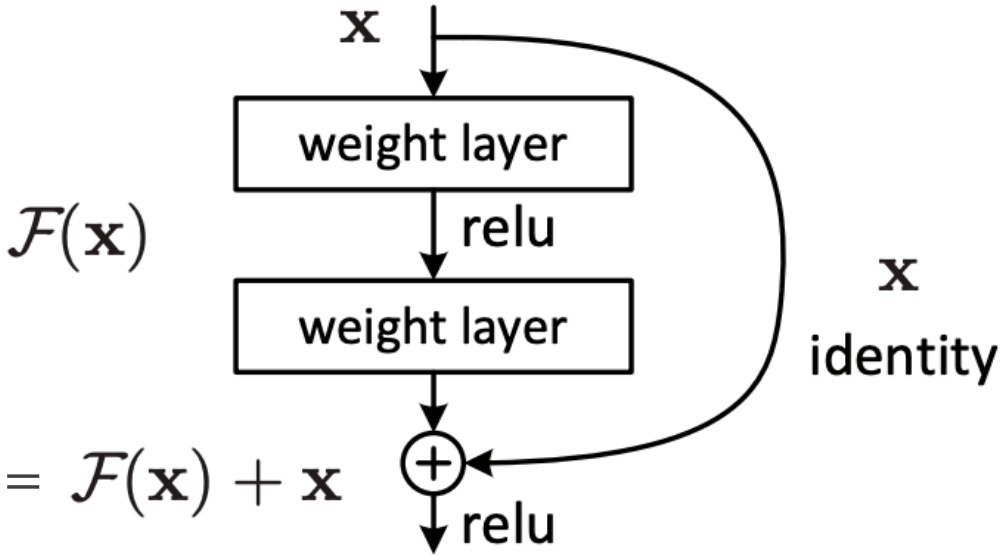
RESNET

- What they noticed: a deeper net (56 layers) was performing worse than a shallower net (20 layers), but not because of overfitting (because it was performing worse even on the training set)
  - Hypothesis: it is because deeper net is a harder optimization problem
- Idea: don't learn the mapping, but learn the residual
  - Residual: what you need to change from the previous representation to achieve the mapping



RESNET

- Learn the residual *F(x)* of the mapping, intead of the complete mapping *H(x)*
- Desirable properties of skip connection :
  - Lets gradient flow back to earlier layers
  - Residual block can easily learn the identity transformation
    - If all weights are zero -> identity transformation



$\mathcal{F}(\mathbf{x})$

$H(x) = \mathcal{F}(\mathbf{x}) + \mathbf{x}$

ANCHORMEN
data activators