9 - CNN architectures

Ludwig Krippahl

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

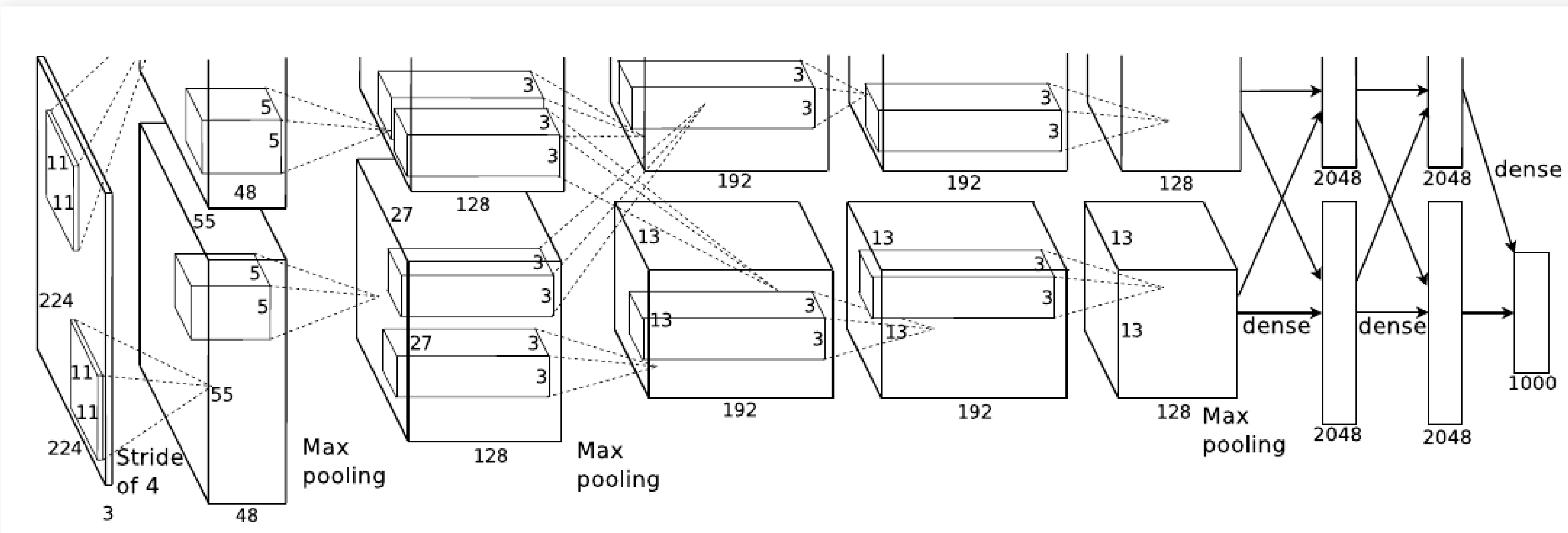
Summary

- CNN architectures for image classification
  - AlexNet, VGG16, ResNet, Inception modules
- CNN architectures for image segmentation
  - FCN, U-Net

Image Classification

AlexNet

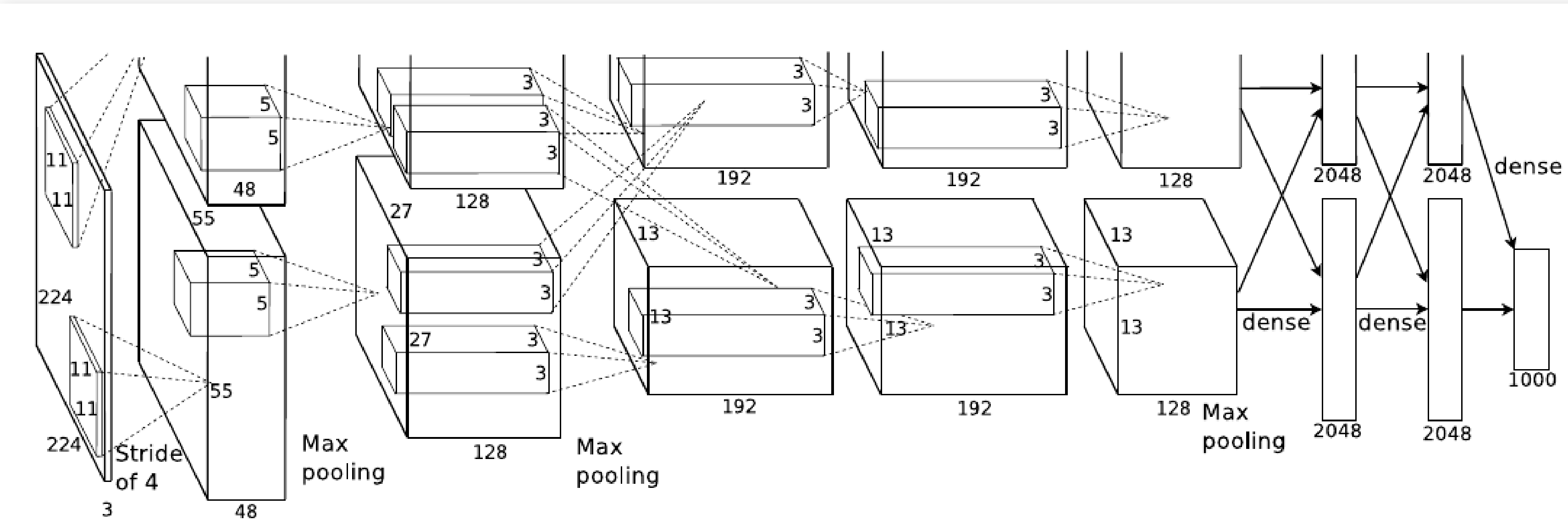
- 60 million trainable parameters
  - (2 graphics cards)
- 11x11, 96 kernels (2x48)
- 5x5, 256 kernels (2x128)
- Then 3x3, dense layers for classification



Krizhevsky, Sutskever and Hinton. Imagenet classification with deep convolutional neural networks.

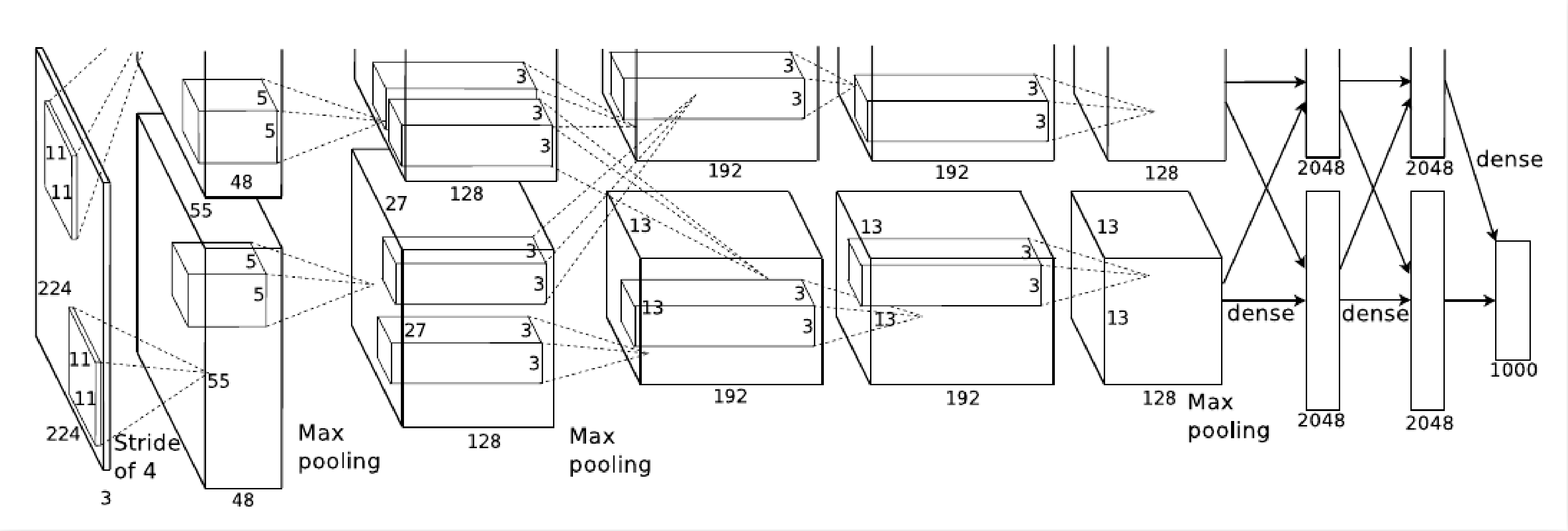
AlexNet

- Convolution layers extract features (patterns)
  - Convolution and pooling makes features less dependent on
- Dense layers for classification, from the extracted features



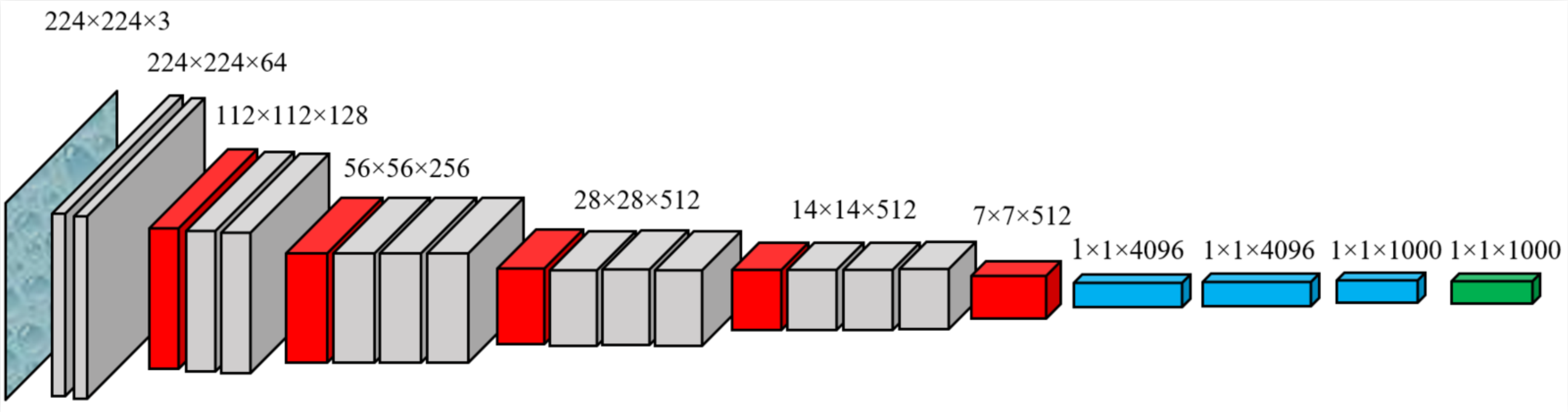
AlexNet

- **One problem:**
  - Large convolution kernels require many weights
  - E.g. 11x11 kernel, 121 weights per input channel



VGG16

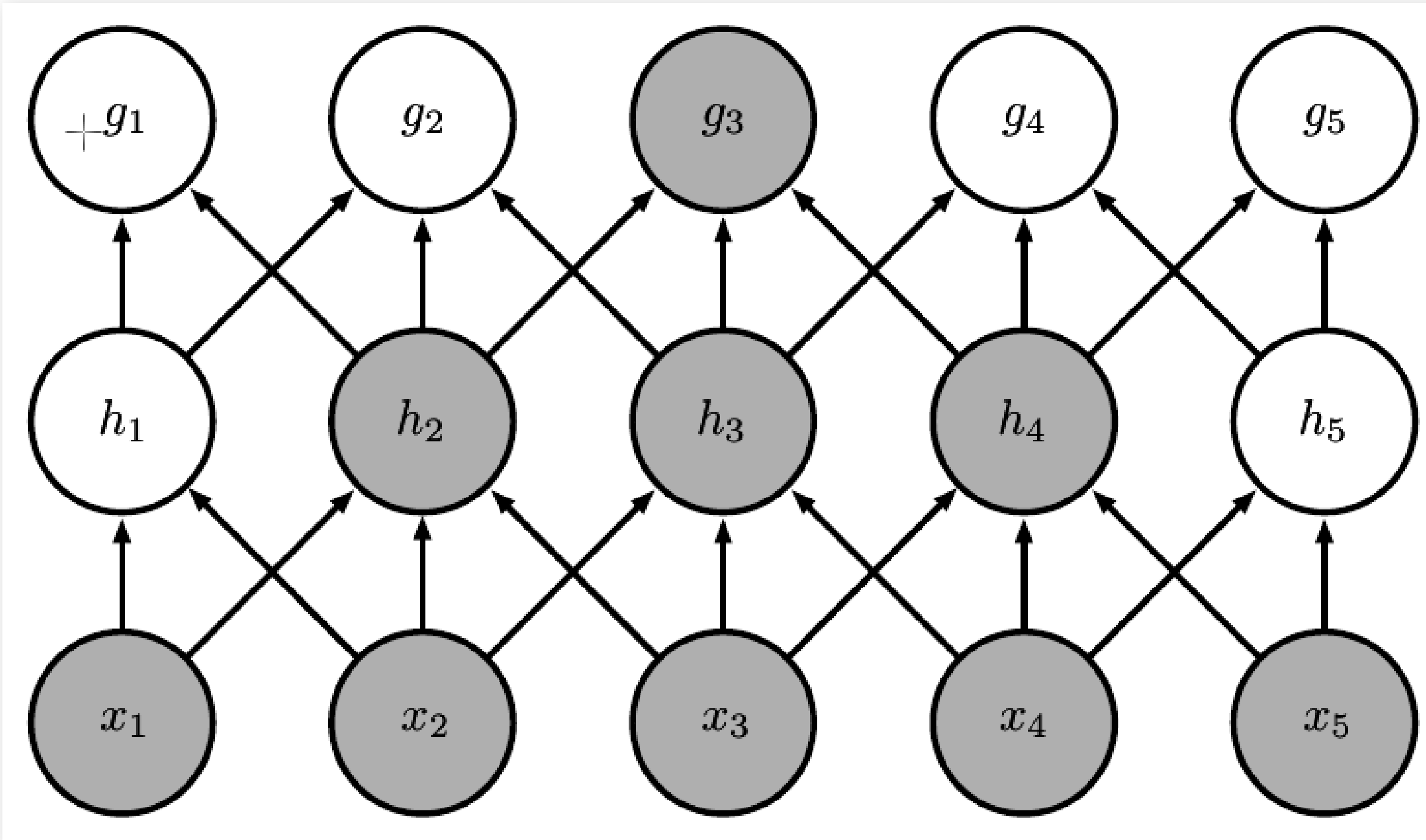
- Red: pooling
- Gray: convolution (stacked)
- Blue: dense
- Green: softmax



Simonyan and Zisserman, Very deep convolutional networks for large-scale image recognition.

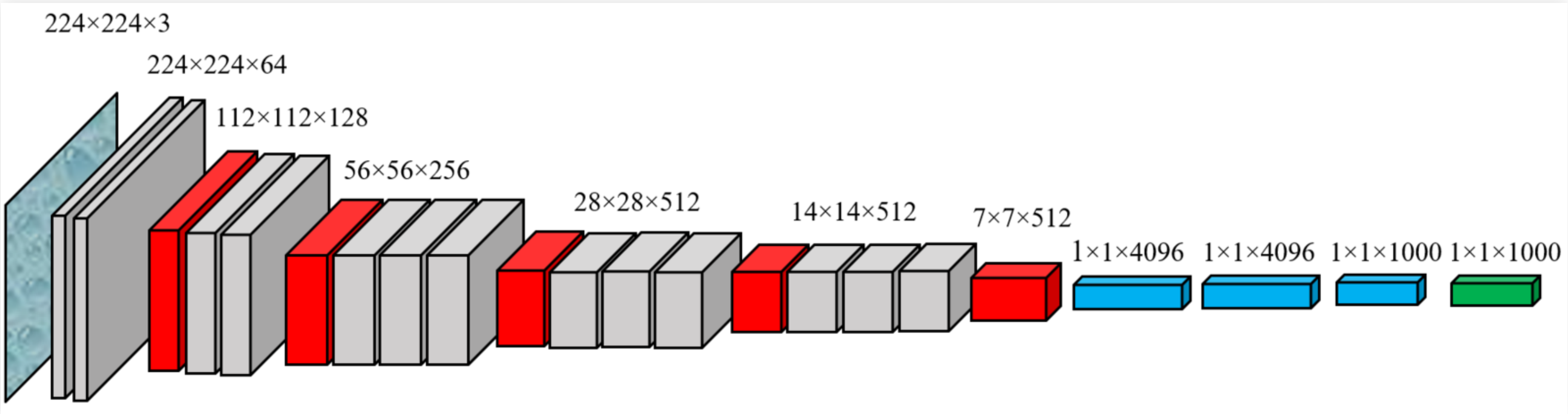
VGG16

- Stacking convolutions



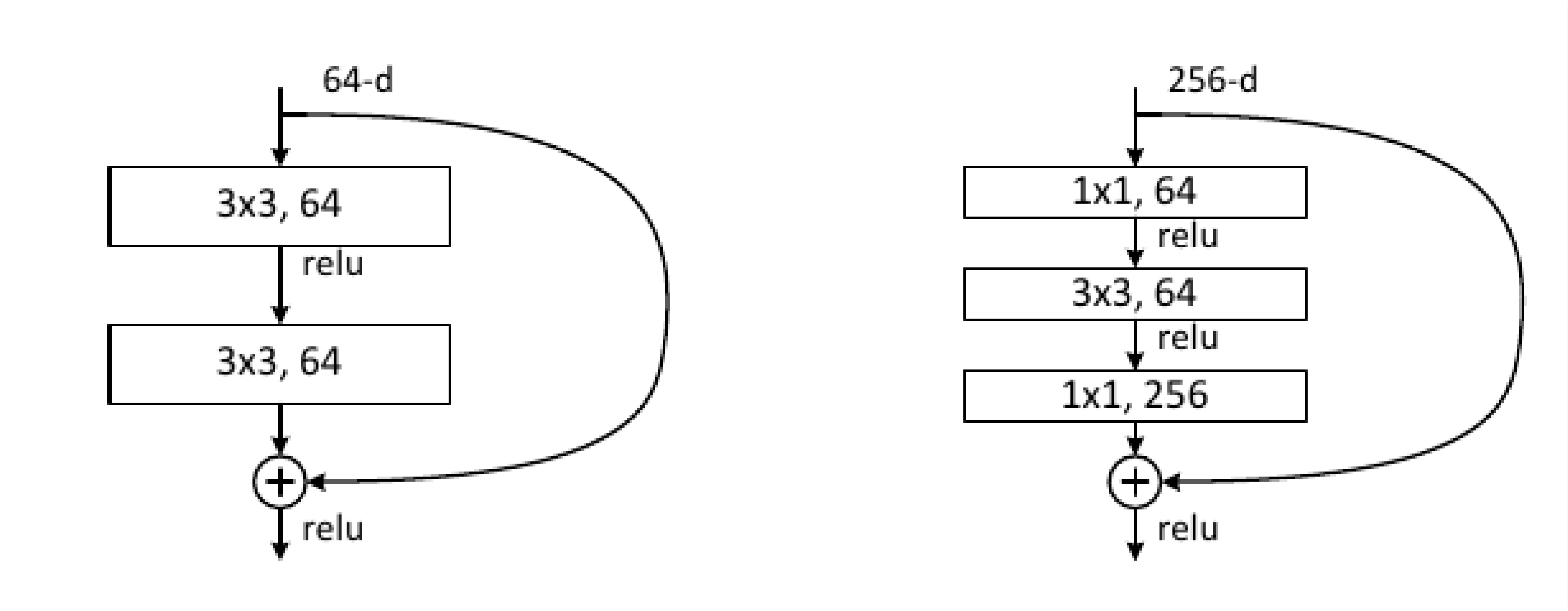
VGG16

- Stacking convolutions
- 5x5 kernel: 25 parameters per channel
  - 3x3 + 3x3 kernels: 9 + 9 = 18 parameters
- 11x11 kernel: 121 parameters per channel
  - five 3x3 kernels; 9 x 5 = 45 parameters per channel



ResNet

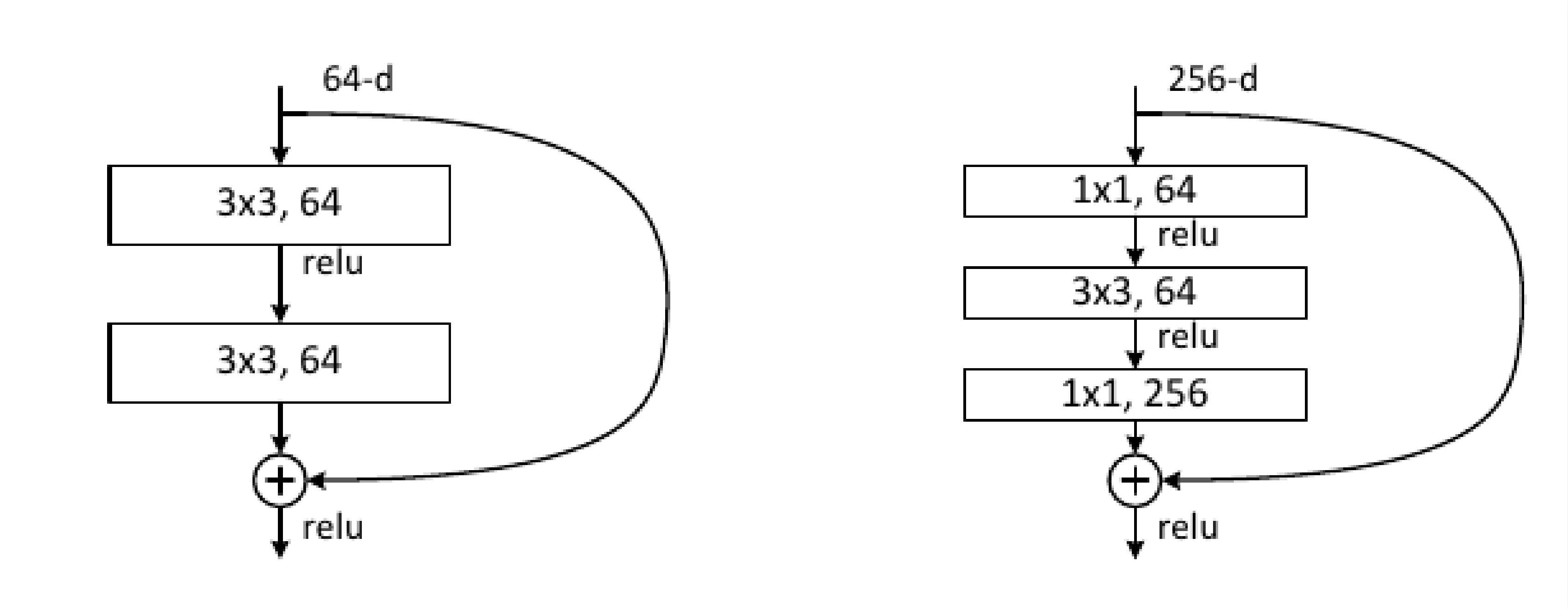
- Deep networks are more powerful
- But they are harder to train
- ResNet makes training easier with residual blocks
  - Layer learns difference (residual) between input and output



He, Zhang, Ren and Sun, Deep residual learning for image recognition.

ResNet

- Residual blocks copy the input to the output
  - Even at the beginning, some useful information passes through
- In addition, ResNet uses 1x1 kernels
  - These output linear combinations of the input channels (plus nonlinear activation)
  - Can be used to change the number of filters



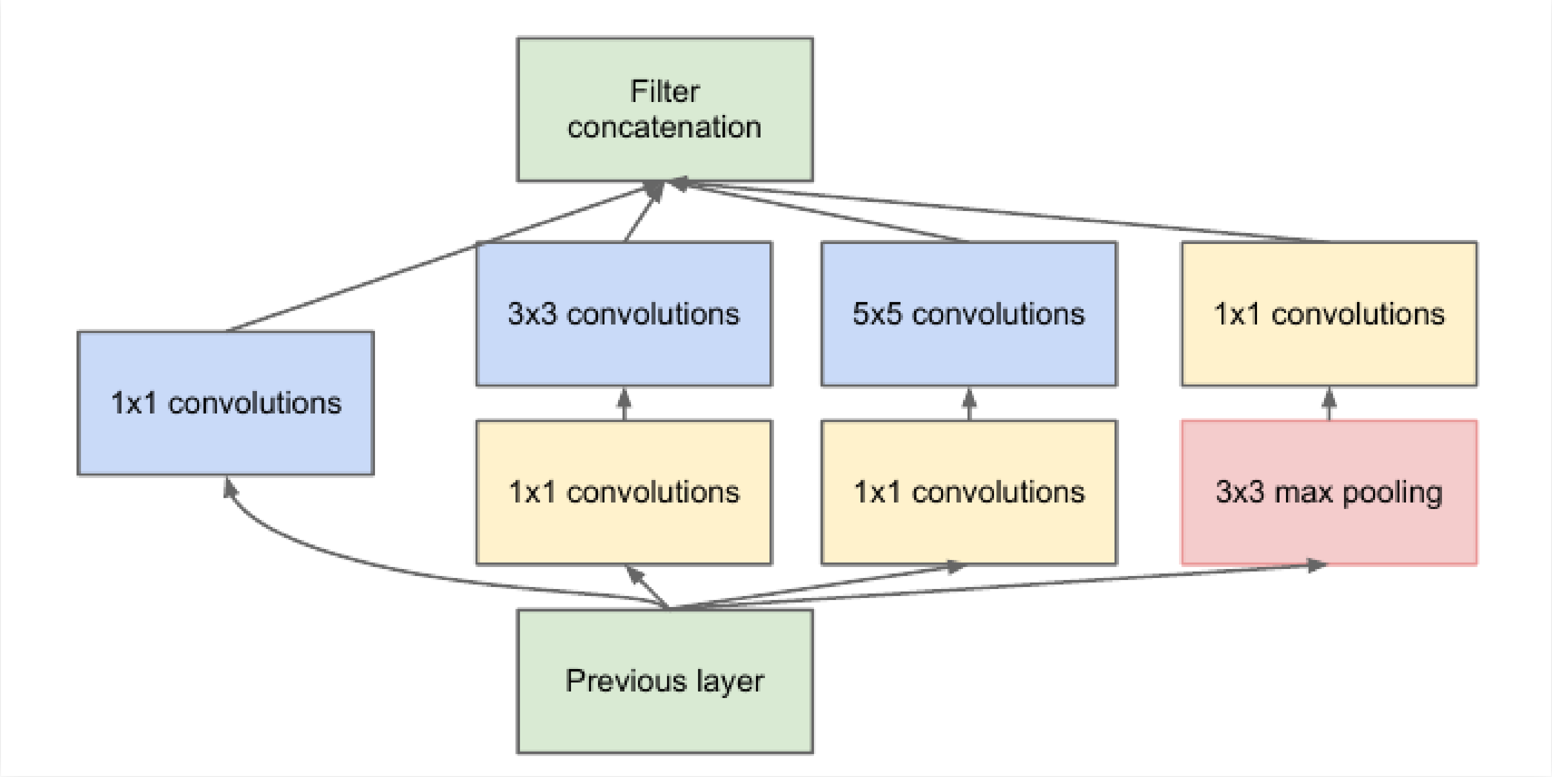
GoogLeNet

Szegedy et al, Going deeper with convolutions.

- Motivation:
  - Larger networks are more powerful
  - But only some combinations of connections seem to be necessary
  - Evidence: dropout works
  - Sparseness is hard to implement
  - Operations with dense matrices are more efficient
- Can we approximate this?
  - Design blocks that can work at different scales
  - But without too many parameters

Inception

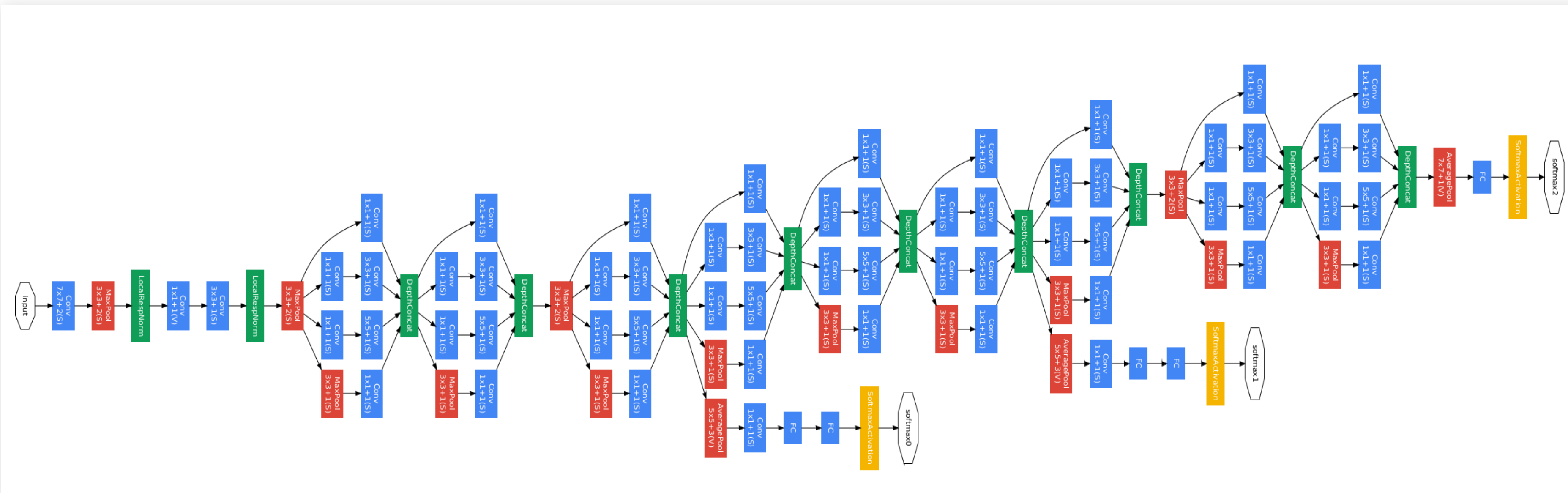
- GoogLeNet uses Inception modules (version 1)
  - All stride 1, same sized feature map, but adjusting number of filters
  - 1x1 kernels: linear combination plus nonlinear activation



Szegedy et al, Going deeper with convolutions.

Inception

- GoogLeNet uses Inception modules (version 1)
  - Stacked with pooling (stride 2) to reduce map size
- Intermediate classifiers contribute to the loss function
  - Did not work as expected, but seem to provide regularization

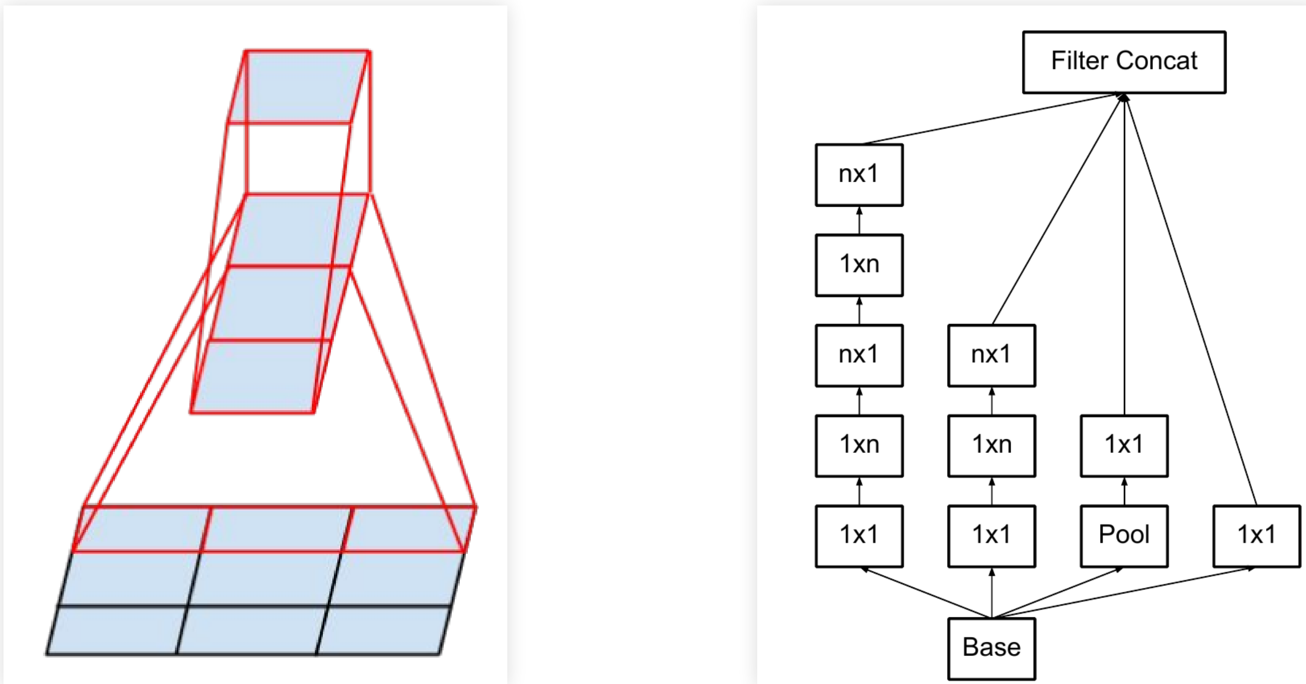


Szegedy et al, Going deeper with convolutions.

Inception

Inception modules (version 2)

- Replace NxN convolutions with stack of 1xN + Nx1
- Uses N = 7 (7x7=49, 7+7=14)

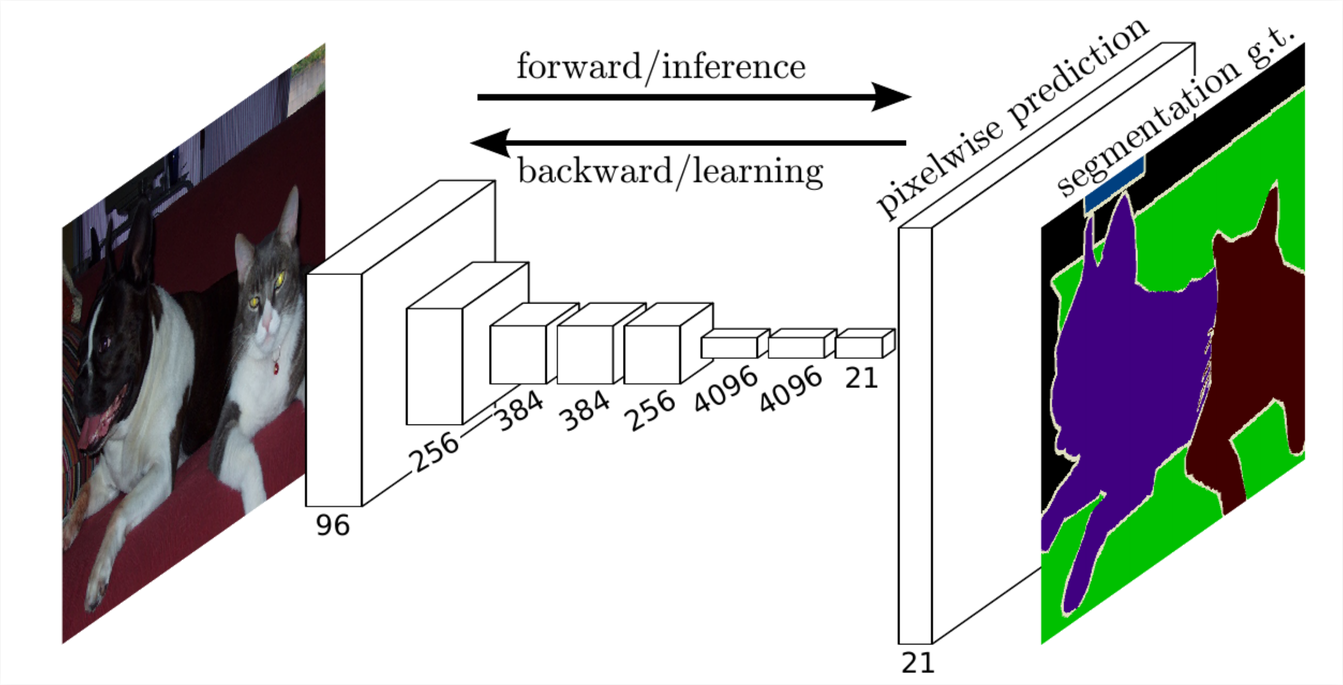


Szegedy et al, Rethinking the inception architecture for computer vision

Image Segmentation

Image Segmentation

- Classify each pixel, output dimensions proportional to input and spatially meaningful



Long, Shelhamer, Darrell, Fully convolutional networks for semantic segmentation.

Image Segmentation

- Two types:
  - Semantic segmentation: distinguish types of object
  - Instance segmentation: distinguish elements

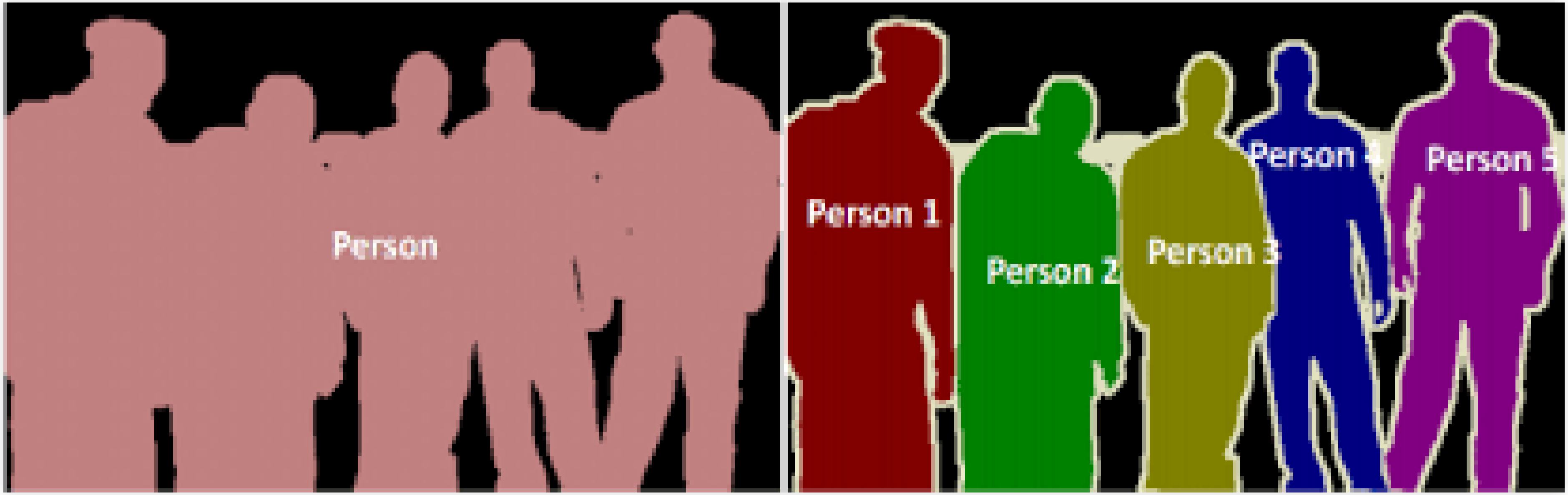


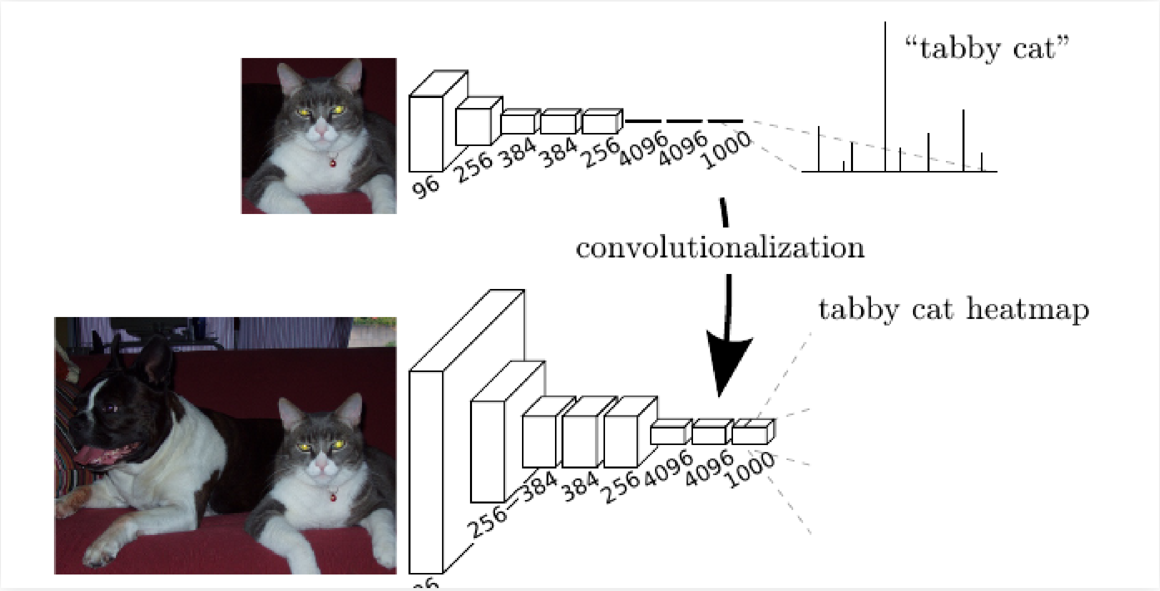
Image from Ross Girshick, Deep Learning for Instance-level Object Understanding

- (we'll cover semantic segmentation)

Fully Convolutional Network (FCN)

Motivation:

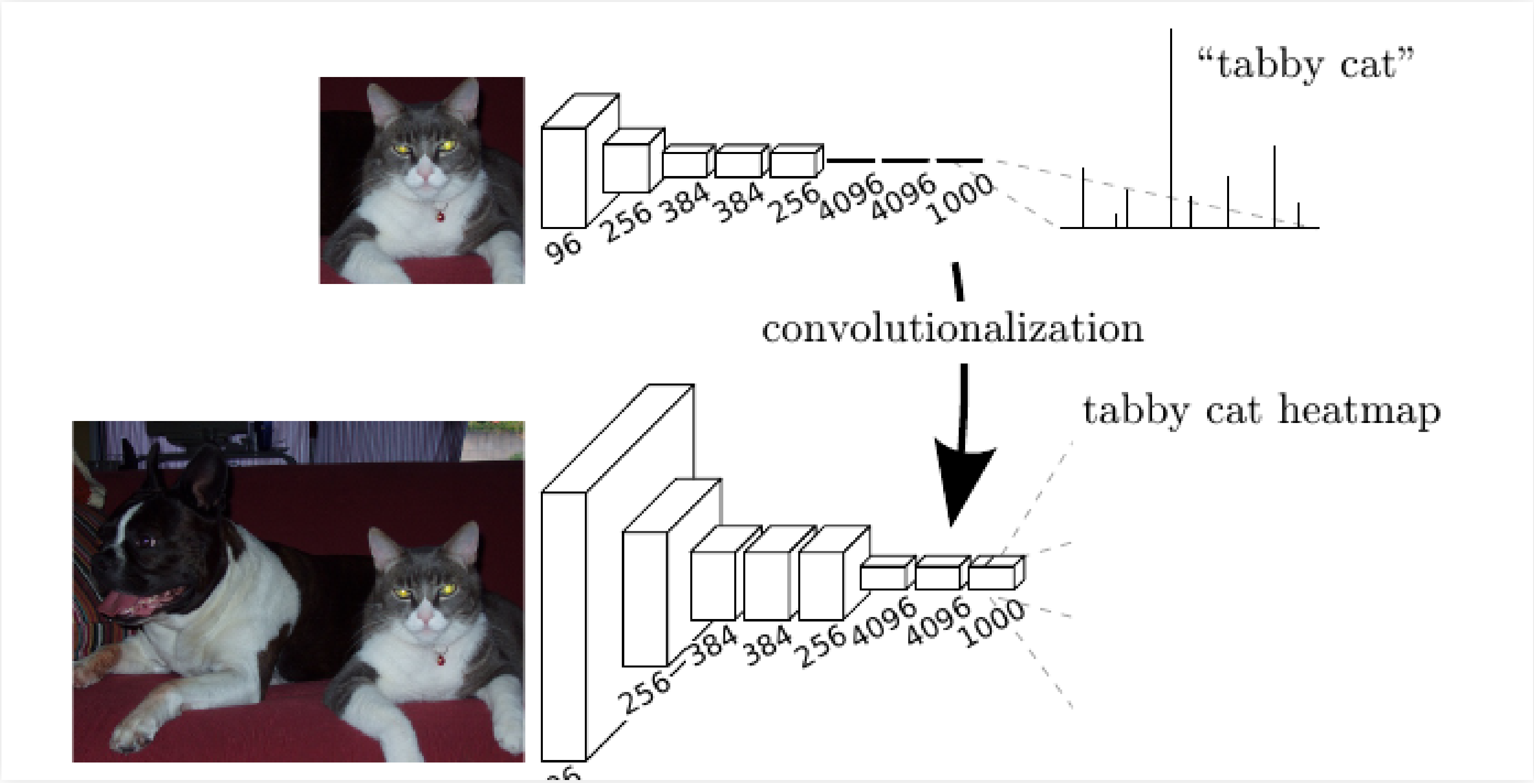
- Classification networks build set of features using convolutions
- Then feed these features into dense layers
- But dense layers are equivalent to convolution with kernel spanning full input
- We can "convolutionalize" classification networks this way



Long, Shelhamer, Darrell, Fully convolutional networks for semantic segmentation.

Fully Convolutional Network (FCN)

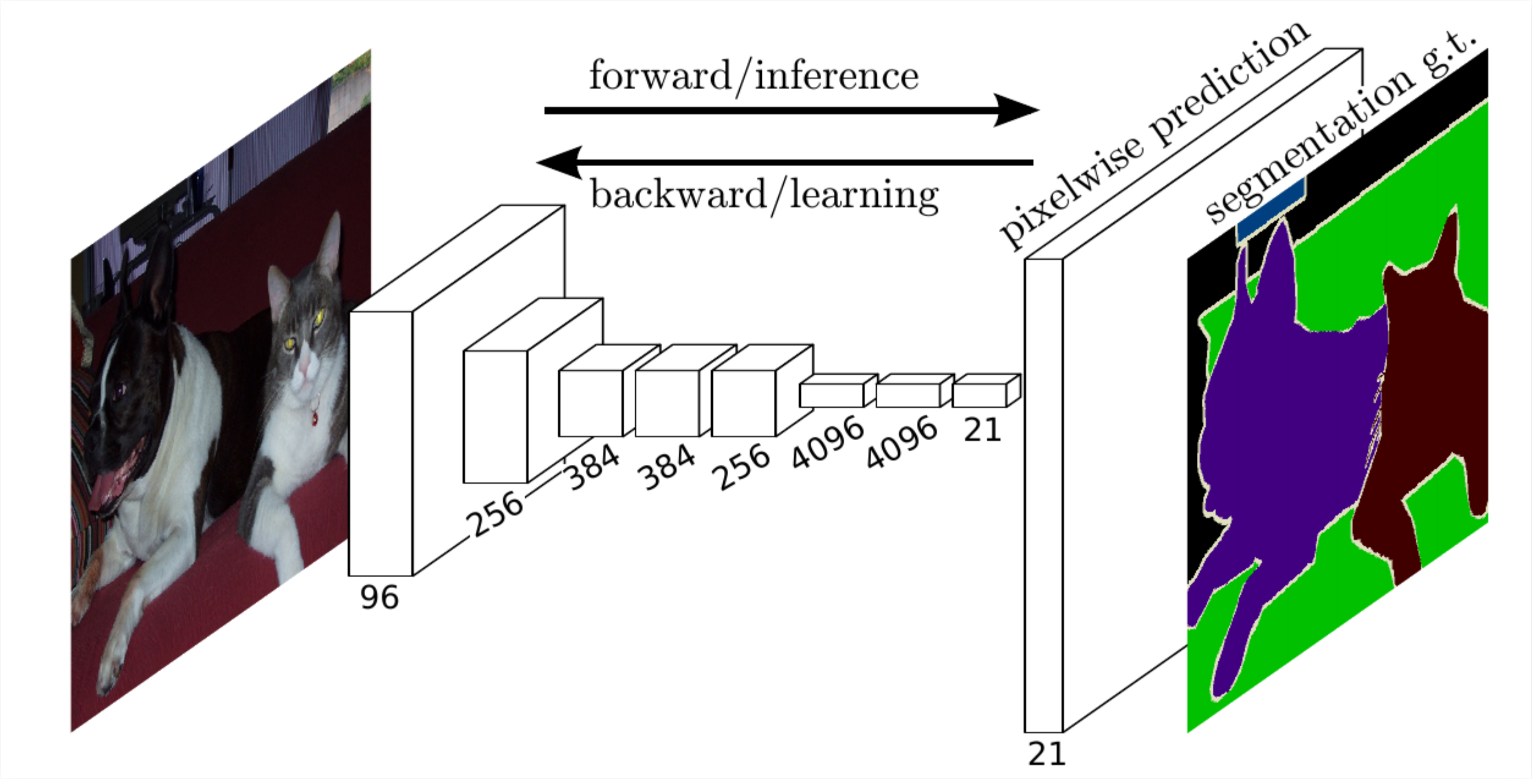
- This gives a set of rough maps
  - For different patches, due to pooling



Long, Shelhamer, Darrell, Fully convolutional networks for semantic segmentation.

Fully Convolutional Network (FCN)

- We can upsample with transposed or fractional stride convolution

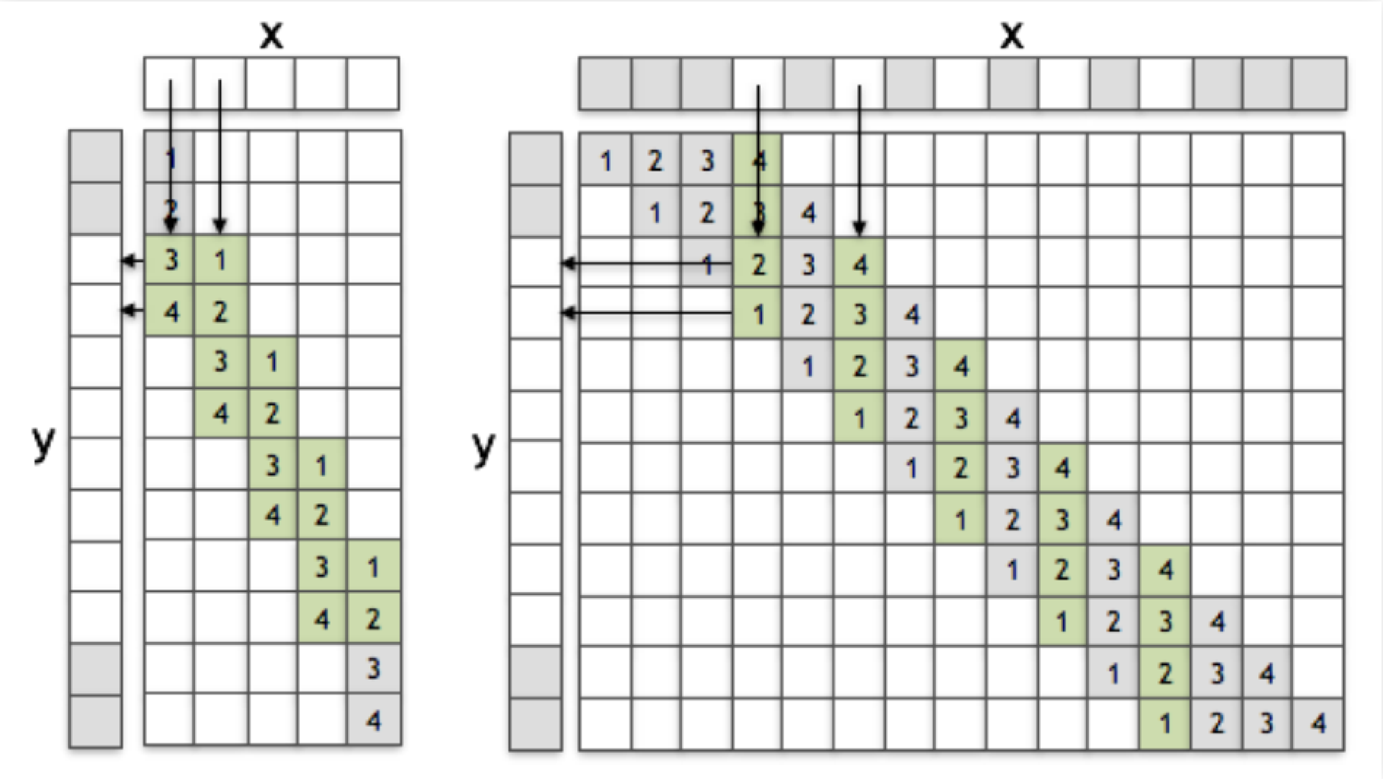


Long, Shelhamer, Darrell, Fully convolutional networks for semantic segmentation.

Fully Convolutional Network (FCN)

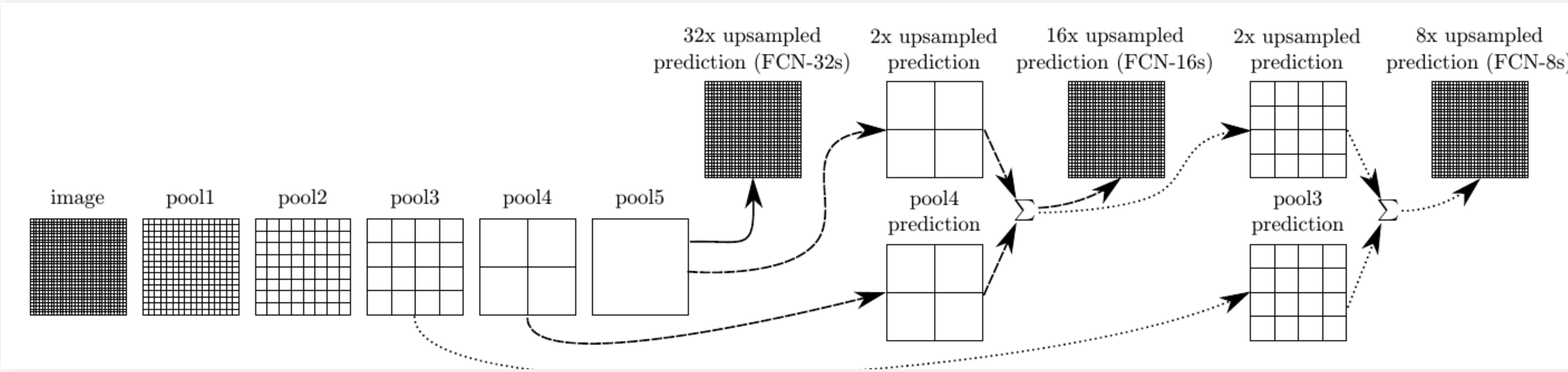
- Transposed and fractional stride convolution

• These convolutions result in output larger than input



Fully Convolutional Network (FCN)

- FCN combines upsampling from different layers

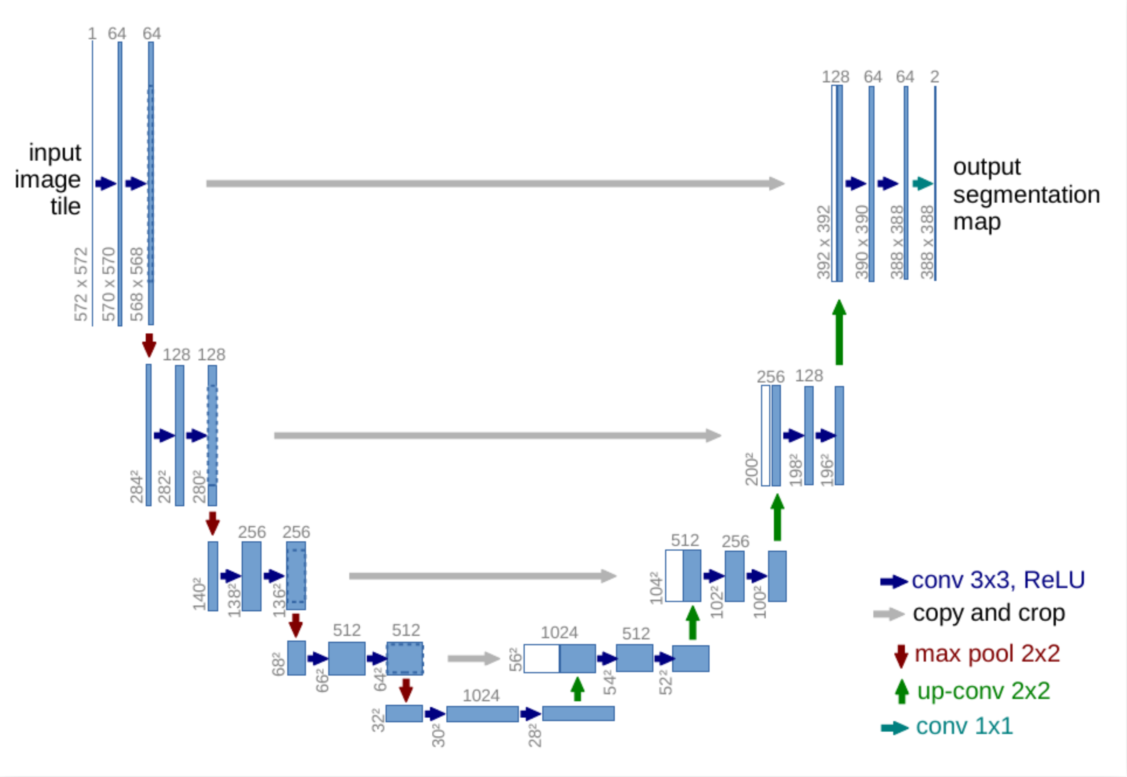


- But it is still one step for upsampling
- And transposed and fractional stride convolution are not ideal
  - (We'll see more on this later, but they cause artefacts due to overlap)

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

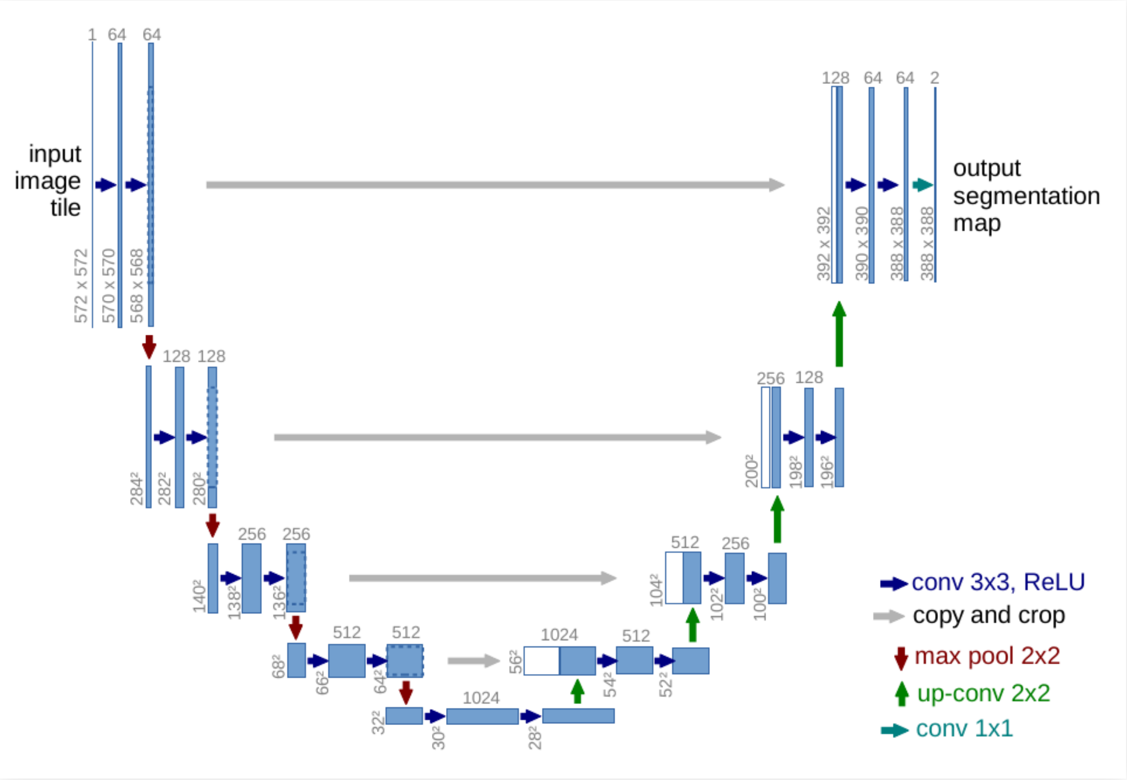
U-Net

- U-Net has a more symmetric profile
  - (Like an autoencoder; more on these next week)



U-Net

- Skip connections to carry feature maps from the contracting part to the expanding part



U-Net

- Does not use transposed or fractional stride convolutions
  - Instead, upsampling (nearest neighbour or interpolation) followed by 2x2 convolution
  - This results in fewer parameters and avoids artefacts
- Loss function:
  - Pixel-wise softmax cross entropy with greater weight for border pixels.

Summary

Summary

- Many different architectures
  - This was just a sample
- Basic ideas:
  - For classification, convolutions followed by dense layers
  - For segmentation, fully convolutional
  - Deeper networks and wider kernels are more powerful
  - More transformations and wider patterns
  - But too many parameters make it harder to train
- Tricks:
  - Residuals and skip connections
  - Decomposing convolutions

Further reading: papers