Deep learning
episode 3, 2022
Convolutional Neural Networks

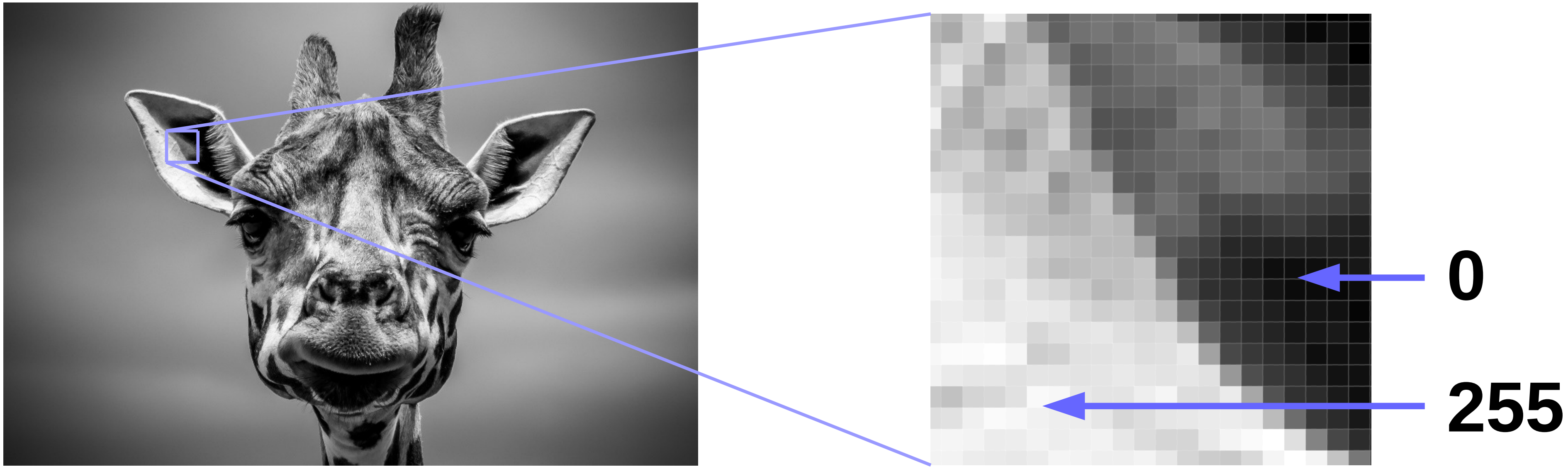
Yandex
Data Factory

LAMBDA

British Hedgehog
Preservation Society

Images

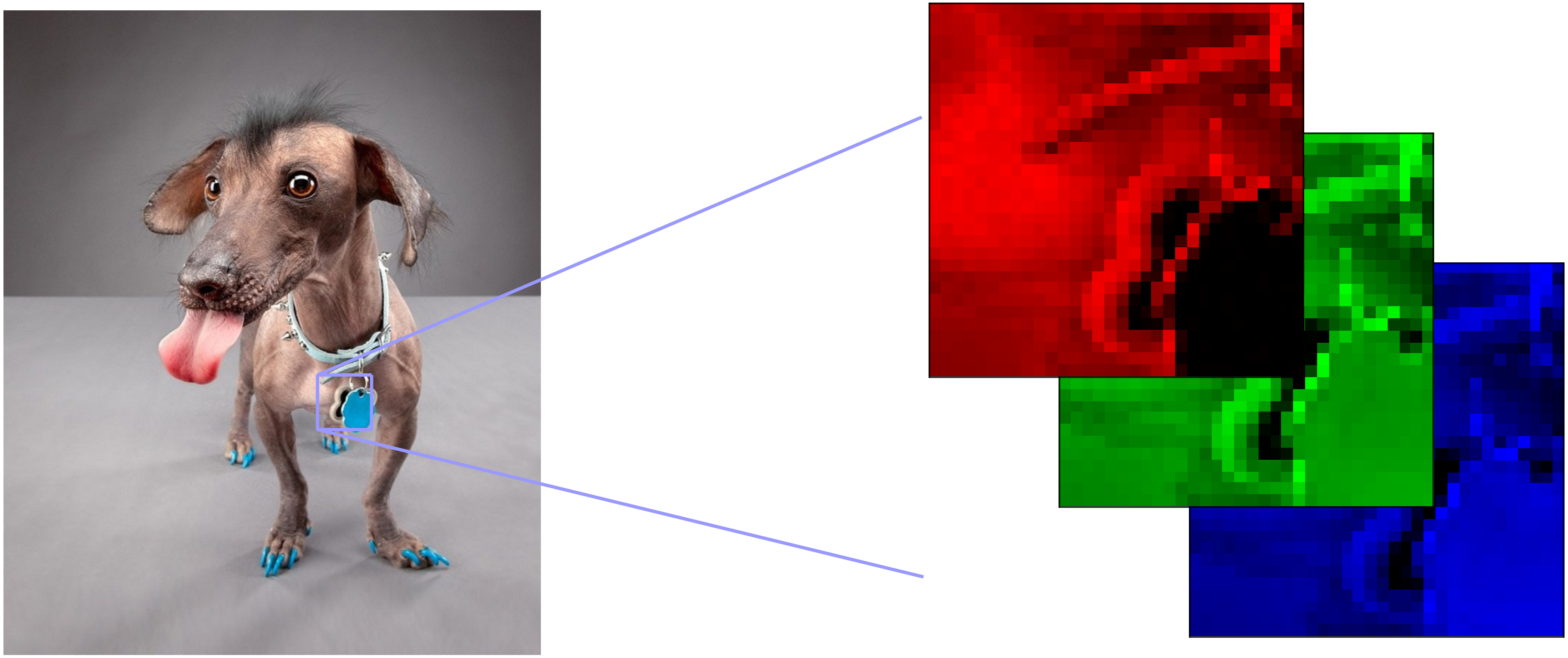
- Grayscale image is a matrix of pixels [H x W]
  - Pixels = **pic**ture **el**ements
- Each pixel stores number [0,255] for brightness



Slide from Andrey Zimovnov's “Deep learning for computer vision”

Images

- RGB image is a 3d array [HxWx3] or [3xHxW]
    - Each pixel stores **R**ed, **G**reen & **B**lue color values [0,255]



Slide from Andrey Zimovnov's “Deep learning for computer vision”

Image recognition

“Dog”

Image recognition

“Gray wall”

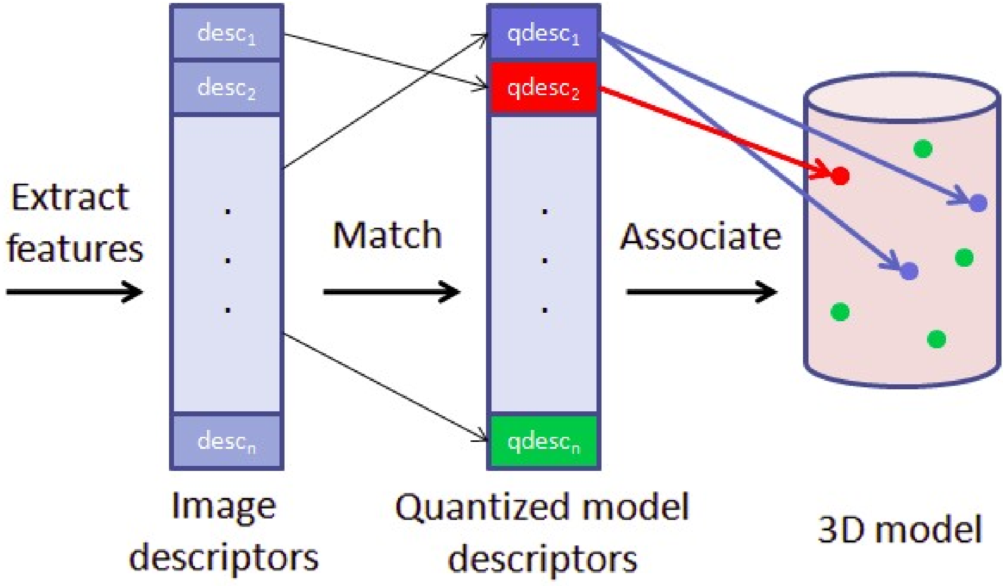
“Dog tongue”

“Dog”

<a particular kind of dog>

“Animal sadism”

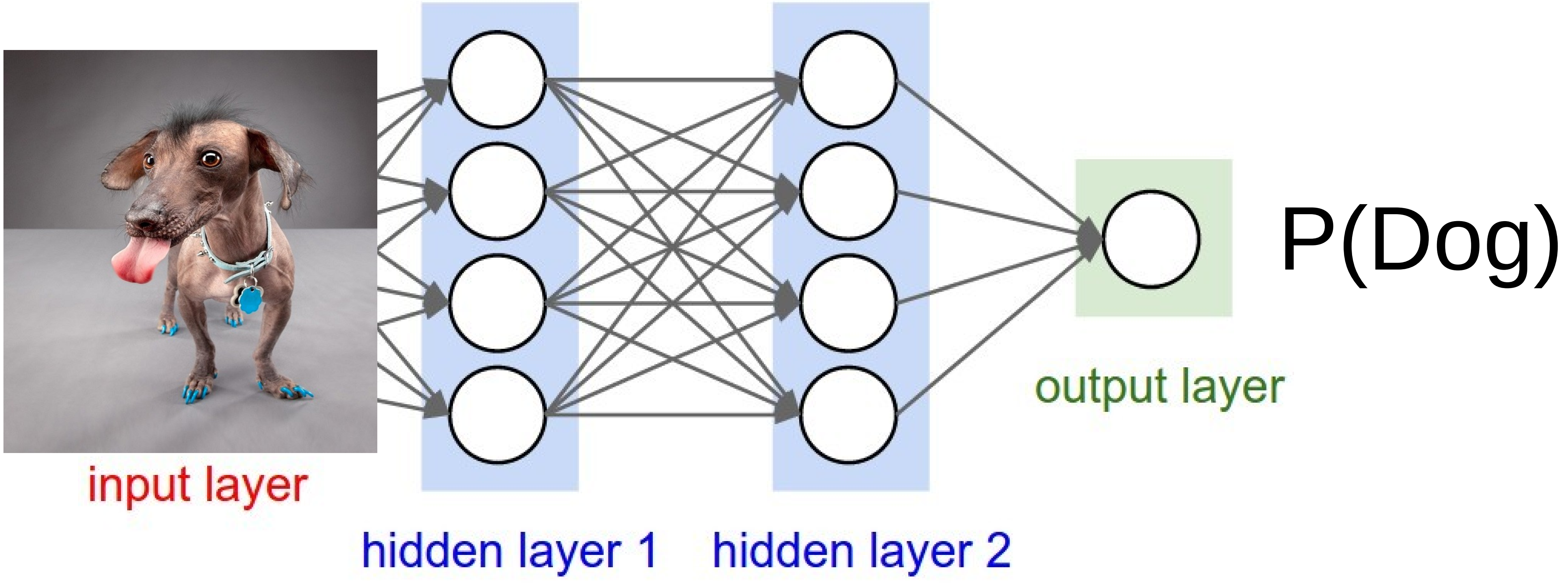
Classical approach



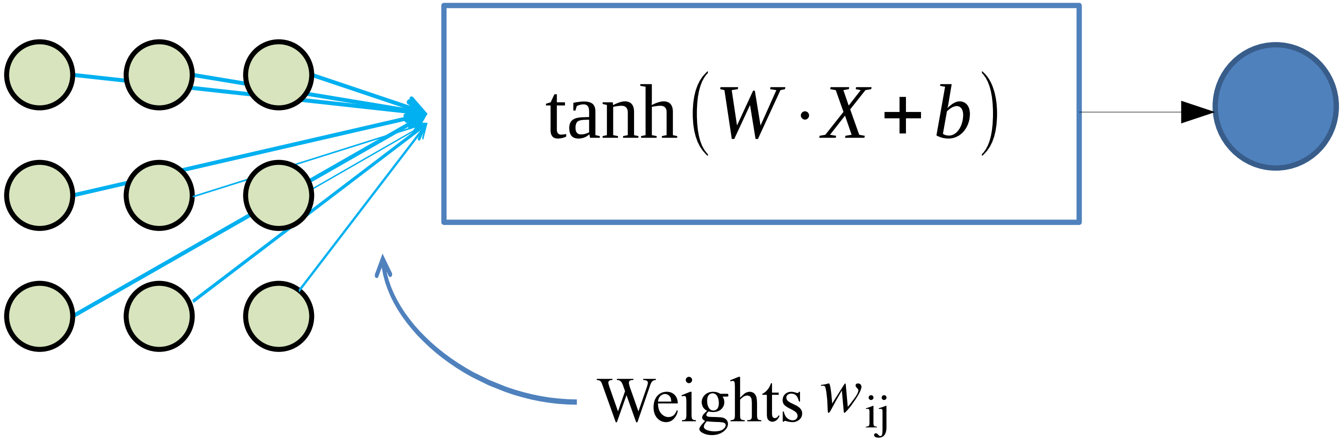
→ "Dog"

Guess what we're going to do now?

NN approach

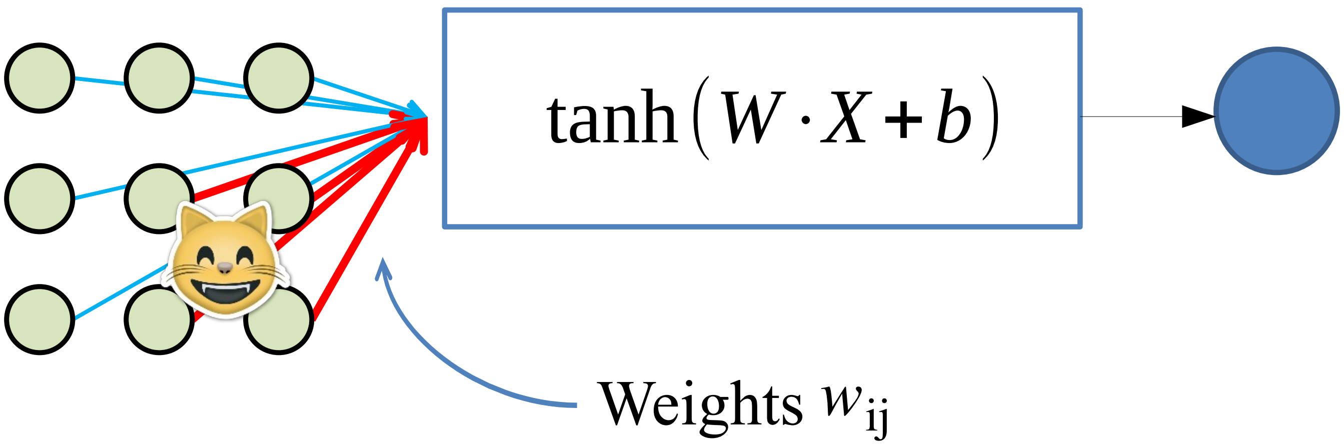


Problem with images



Slide from Andrey Zimovnov's “Deep learning for computer vision”

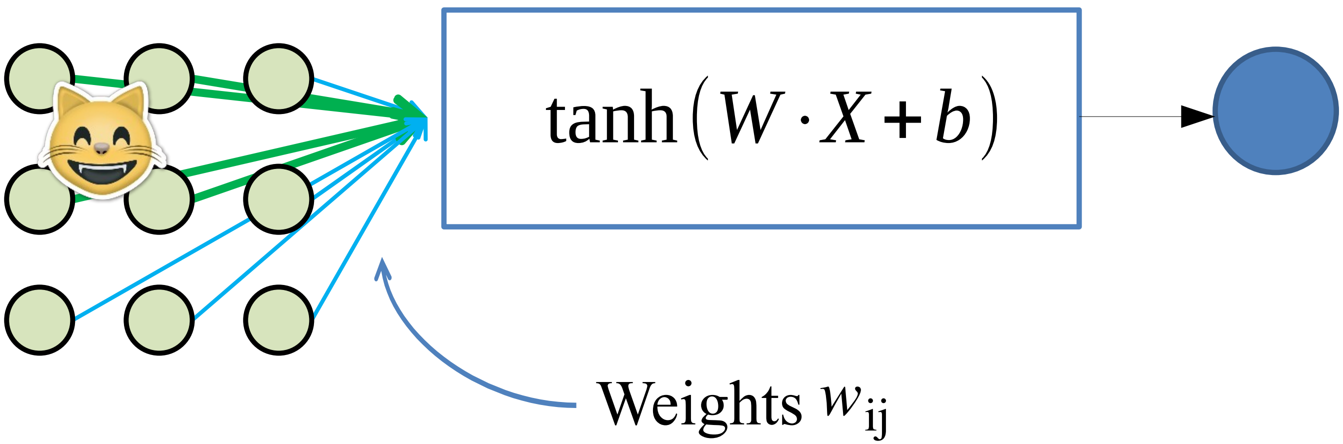
Problem with images



On this object, you will train red weights to react on cat face

Slide from Andrey Zimovnov's “Deep learning for computer vision”

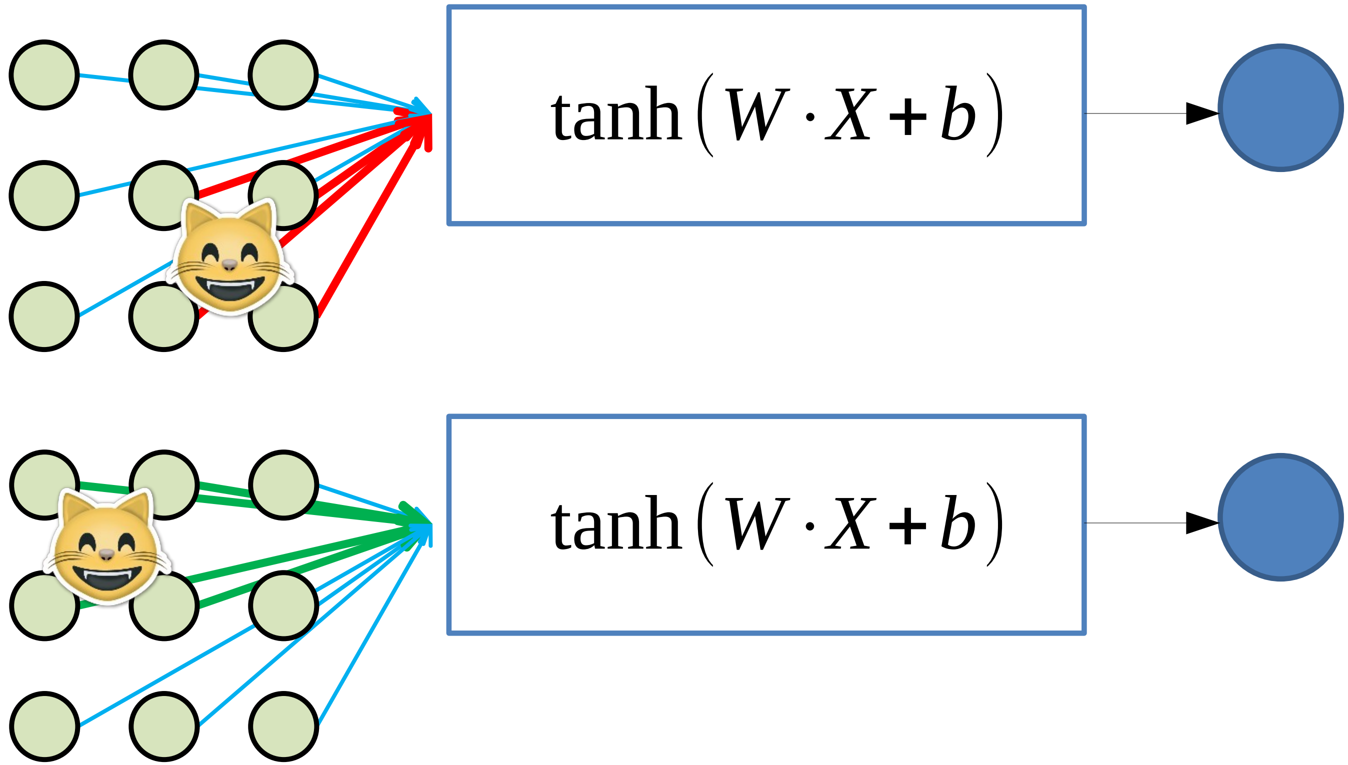
Problem with images



On this object, you will train green
weights to react on cat face

Slide from Andrey Zimovnov's “Deep learning for computer vision”

Problem with images



You network will have to learn those two cases separately!
Worst case: one neuron per position.

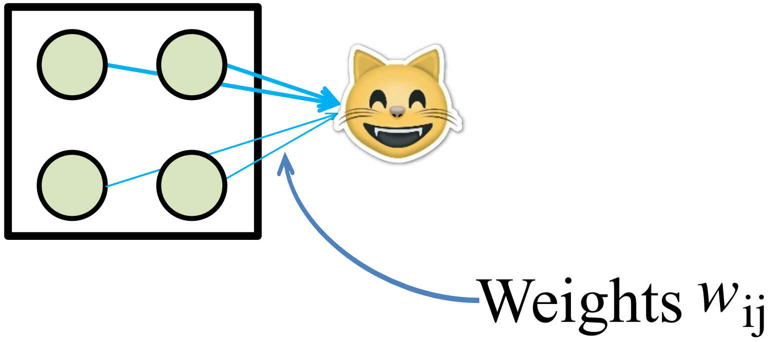
Slide from Andrey Zimovnov's “Deep learning for computer vision”

Problem

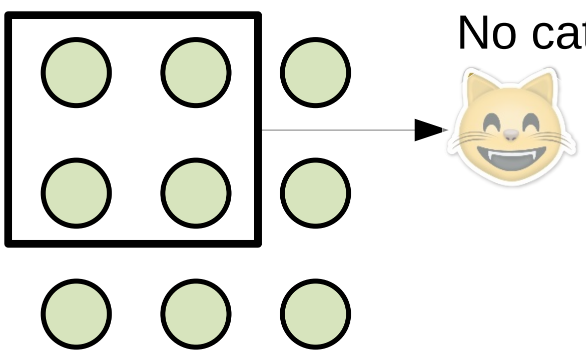
Idea: force all these “cat face” features
to use **exactly same weights**,
shifting weight matrix each time.

Same features for each spot

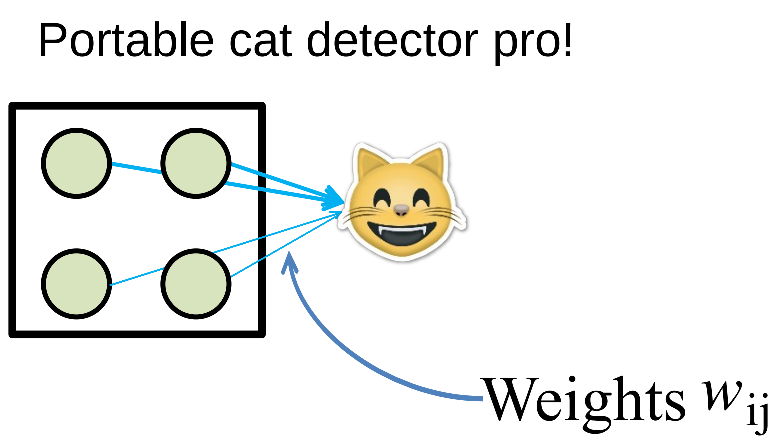
Portable cat detector pro!

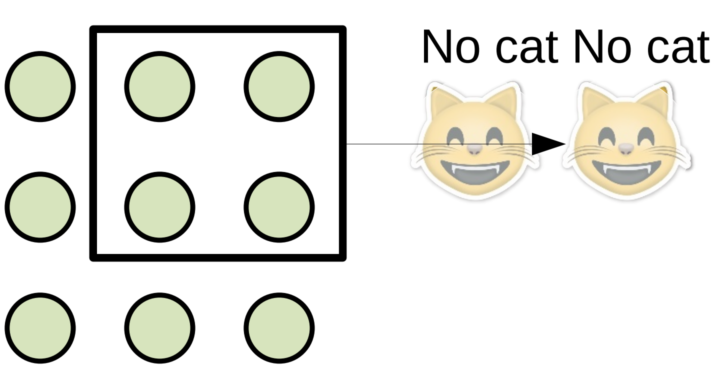


No cat

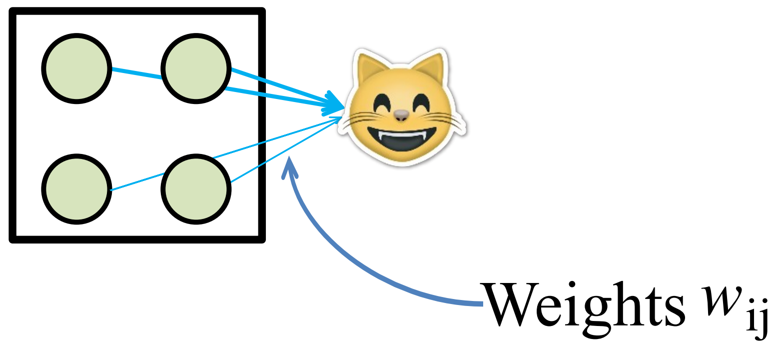


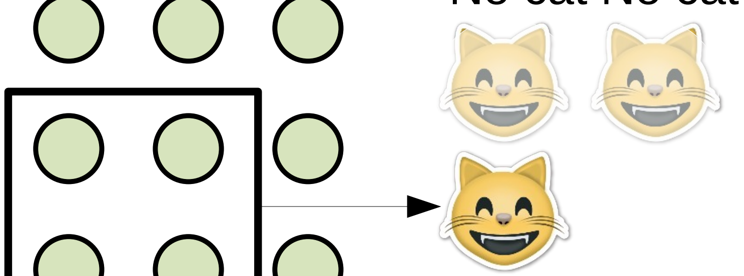
Same features for each spot



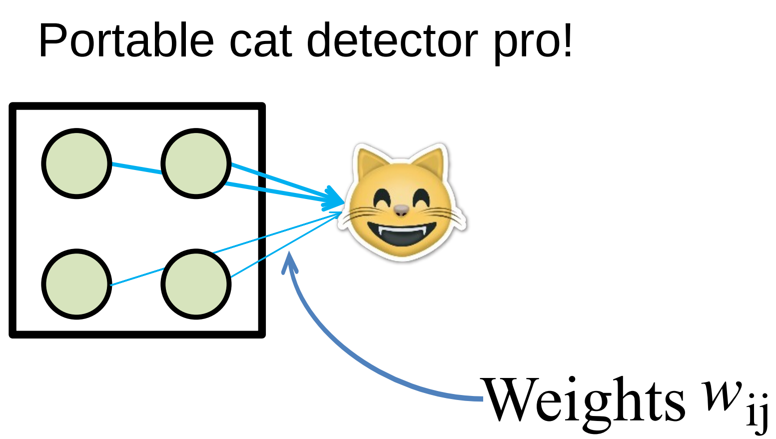


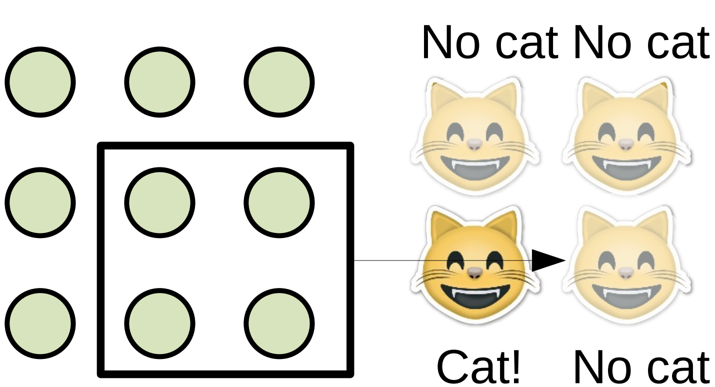
Same features for each spot





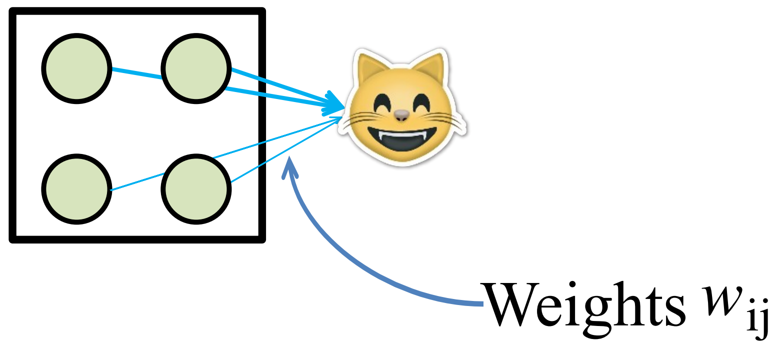
Same features for each spot

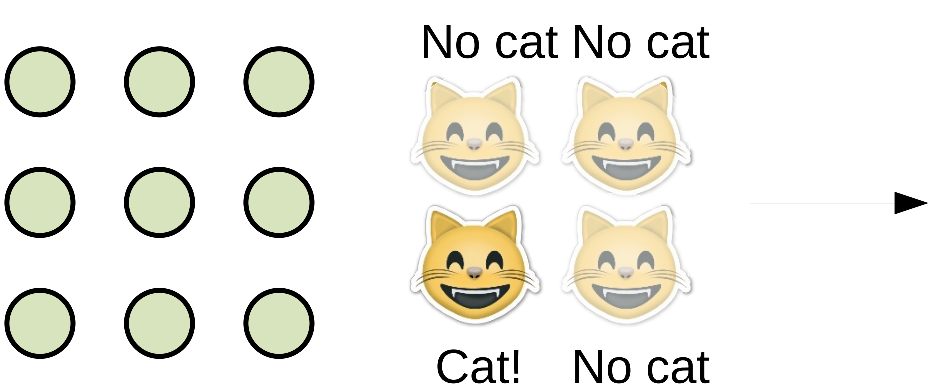




Same features for each spot

Portable cat detector pro!

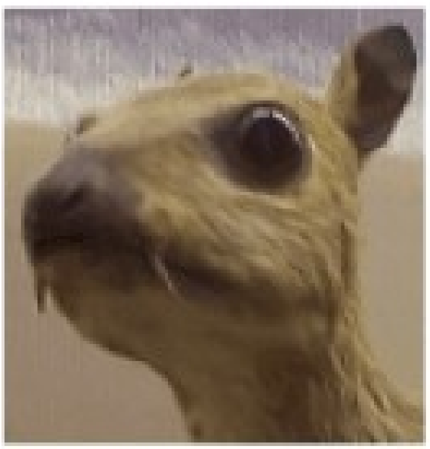




There's a cat here!

Convolution

Input image



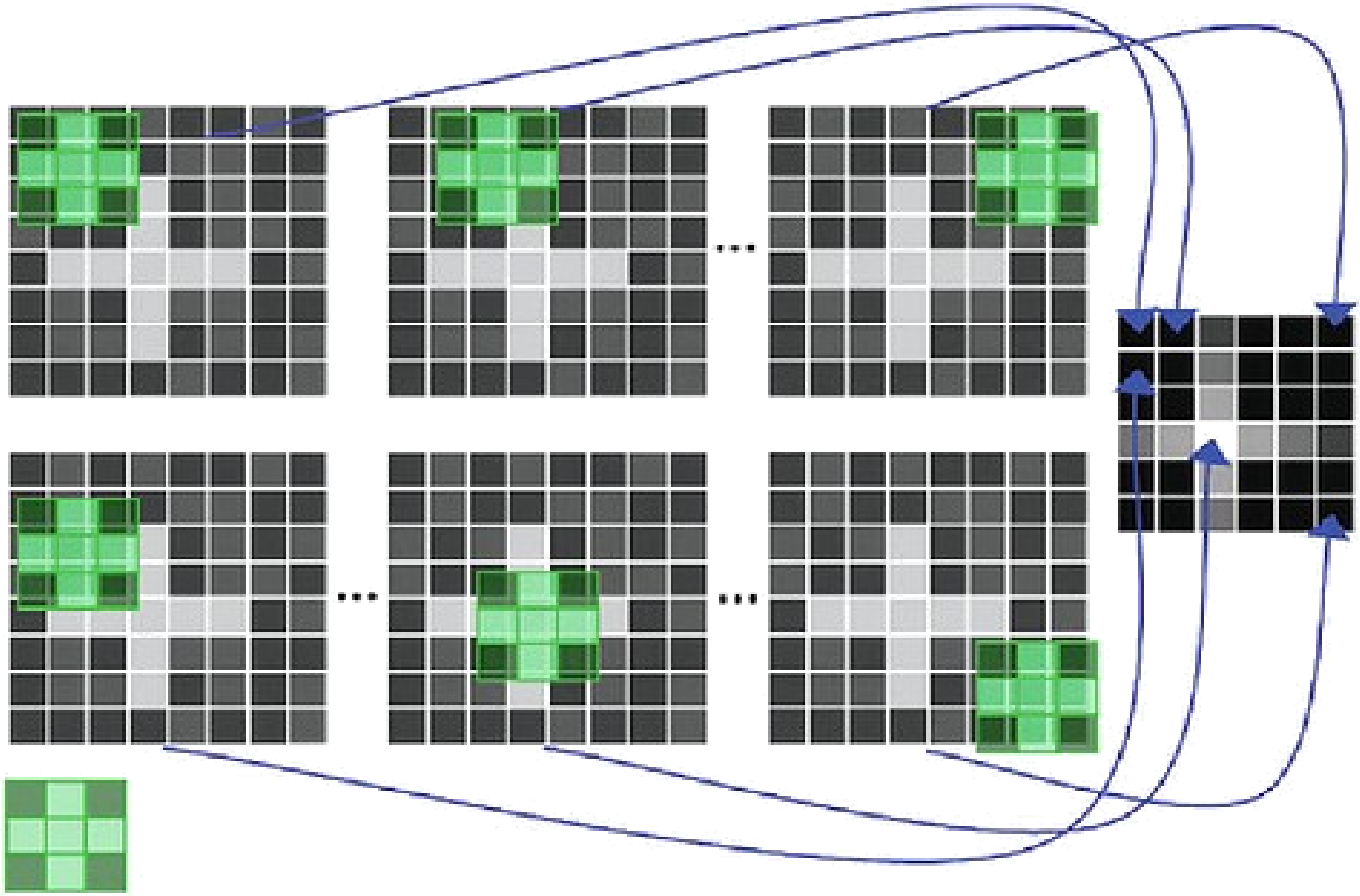
Convolution Kernel

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Feature map

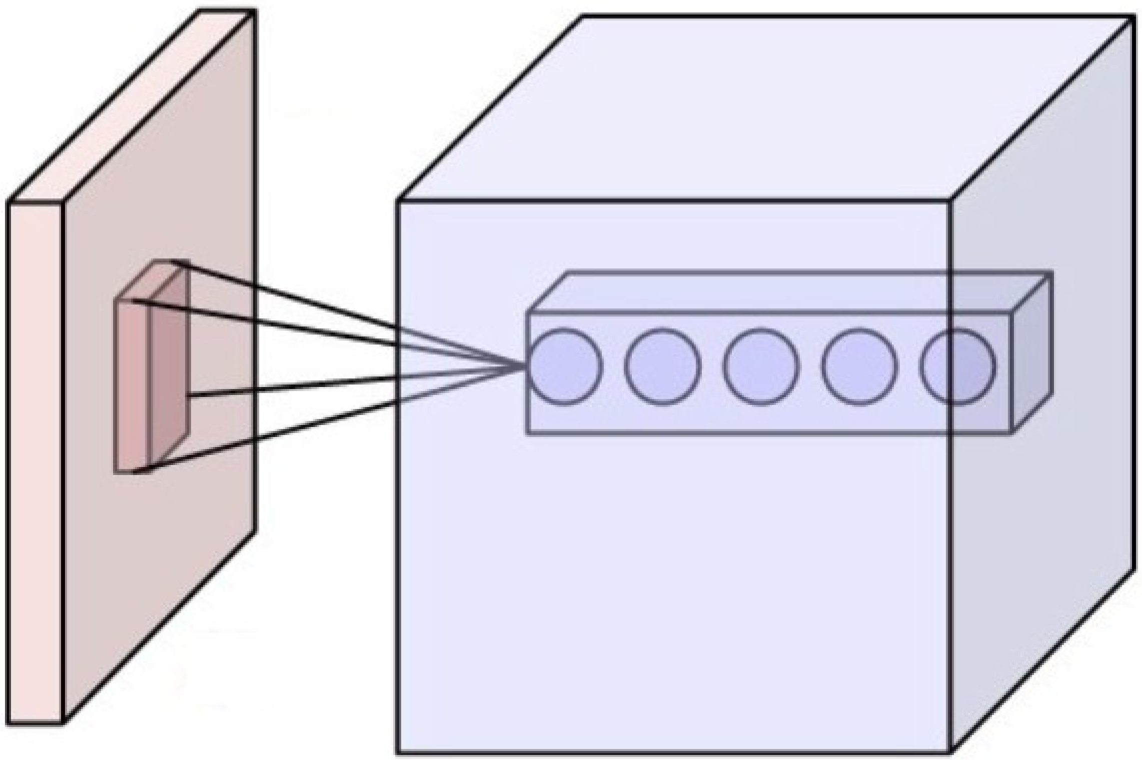
Intuition: how **edge-like** is this square?

Convolution



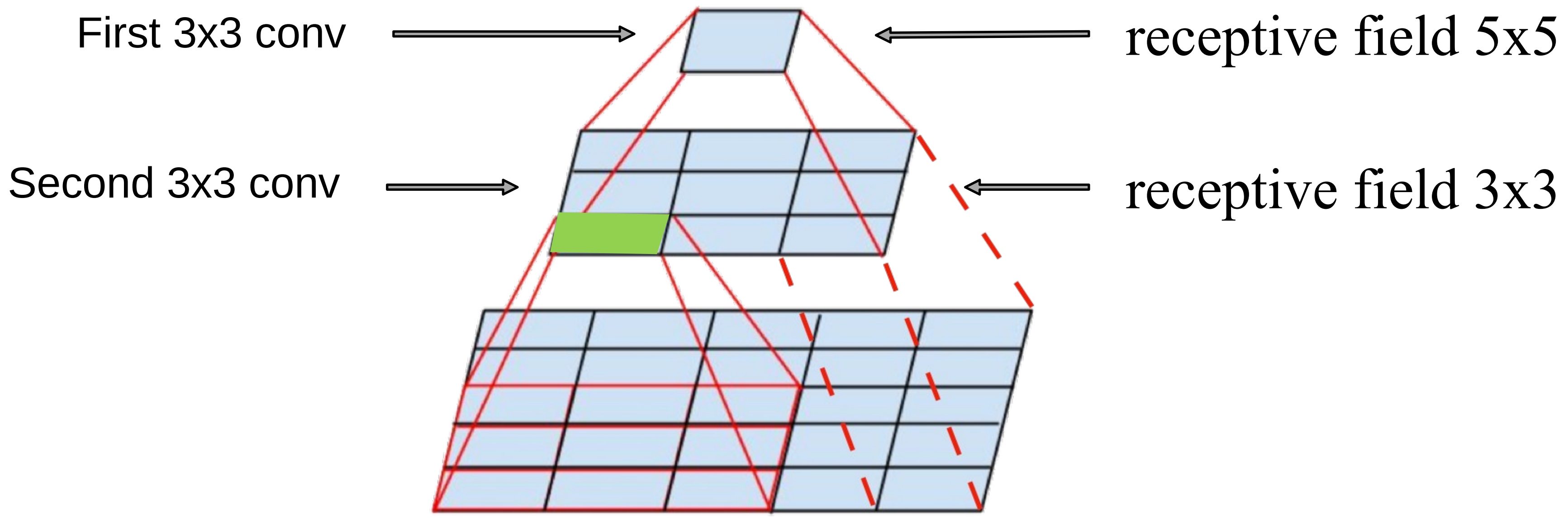
apply same filter to all patches

Convolution



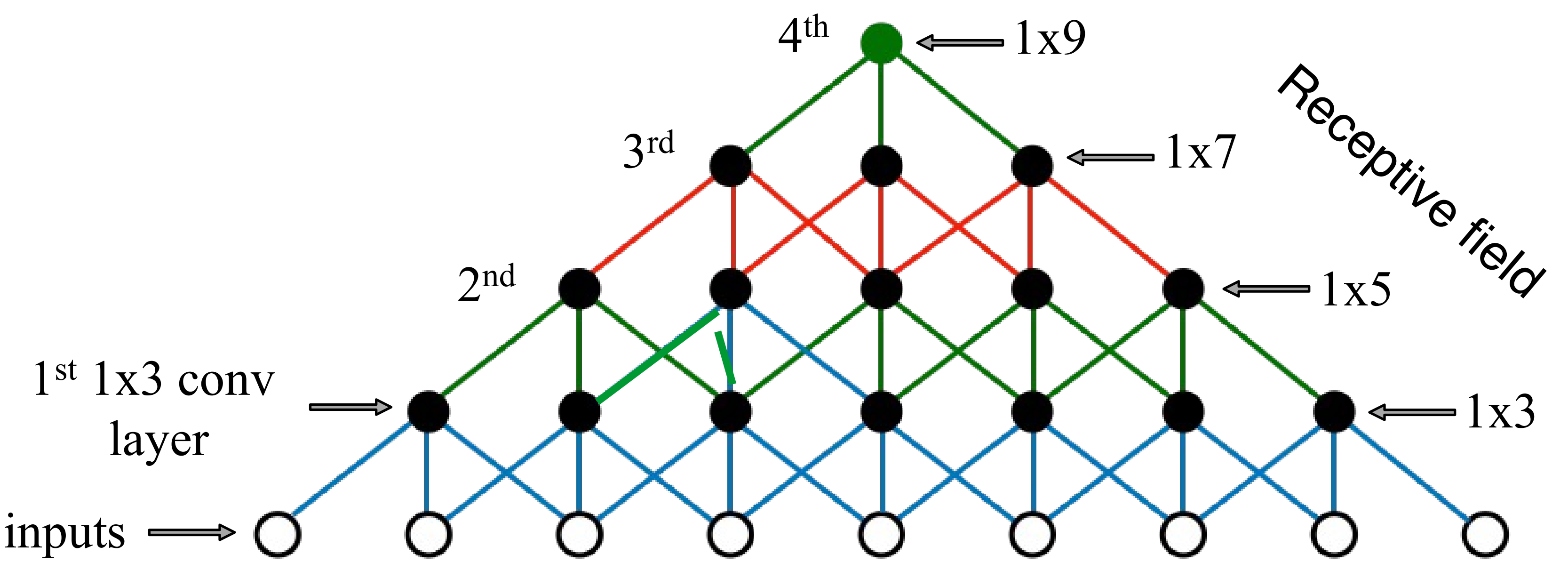
Intuition: how cat-like is this square?

Receptive field



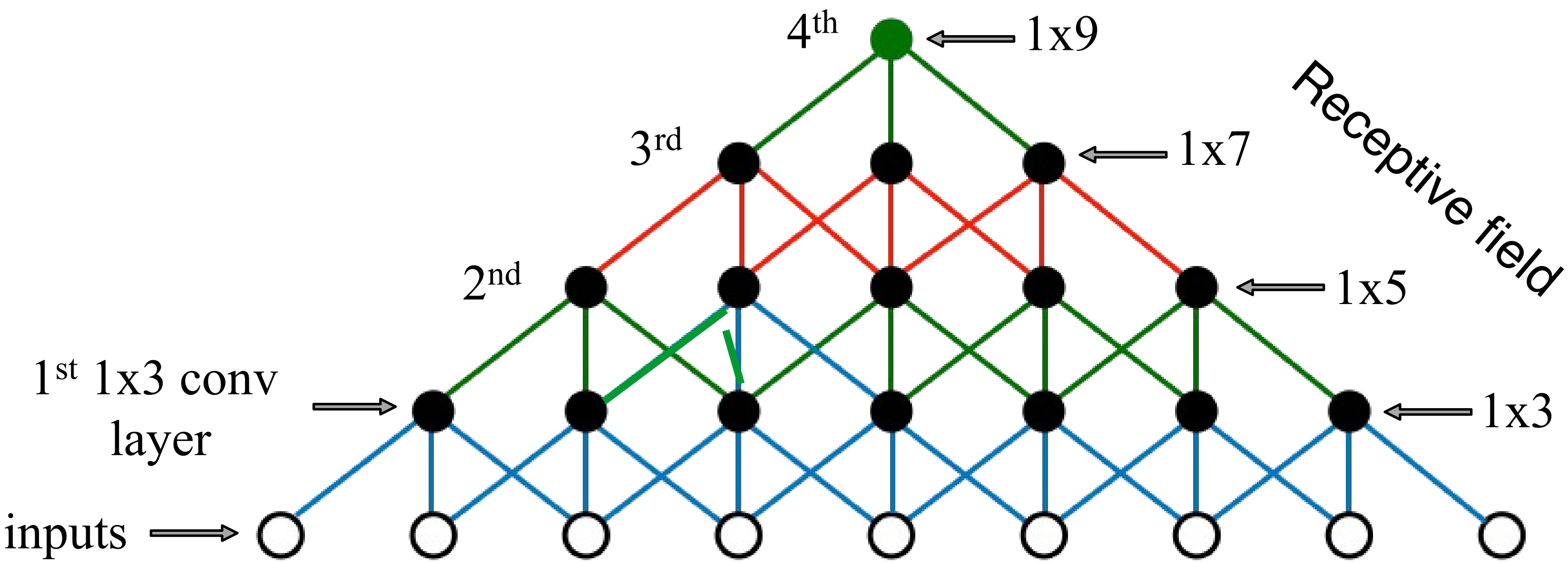
We can recognize larger objects by stacking several small convolutions!

Receptive field



Q: how many 3x3 convolutions we should use to recognize a 100x100px cat

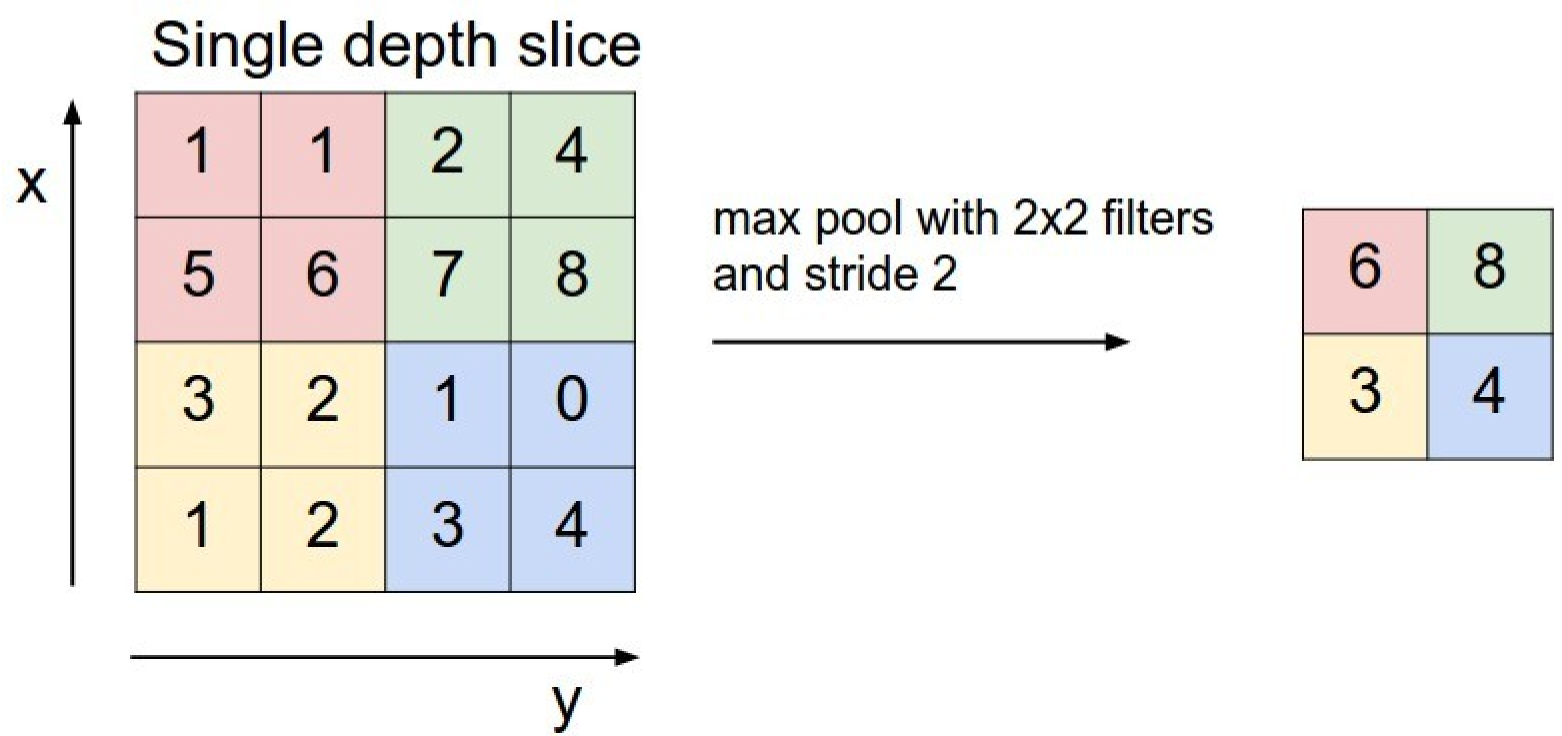
Receptive field



Q: how many 3x3 convolutions we should use to recognize a 100x100px cat

A: around 50... we need to increase receptive field faster!

Pooling



Intuition: What is the max cat-likelihood over this area?

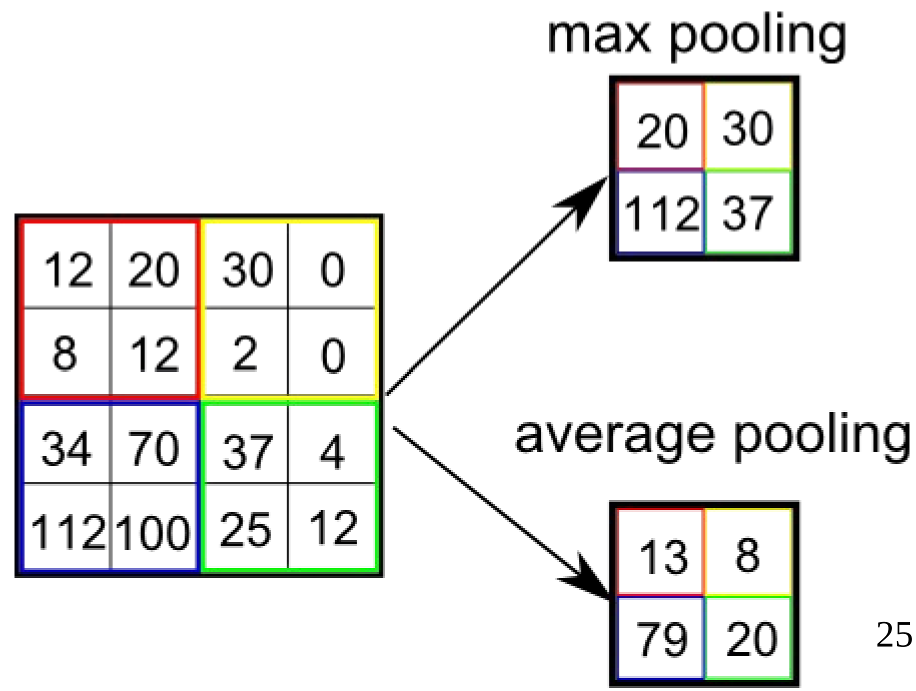
Pooling

Motivation:

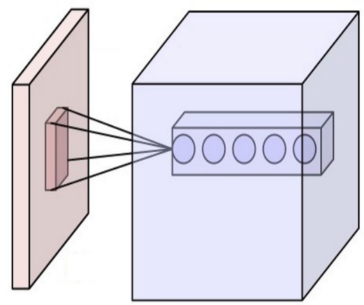
- Reduce layer size by a factor
- Make NN less sensitive to small image shifts

Popular types:

- Max
- Mean(average)



Convolution

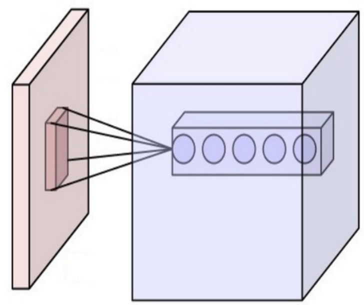


?!

Filters: 100x(3x5x5)

Image : 3 (RGB) x 100 px x 100 px

Convolution



100x96x96

~10^6

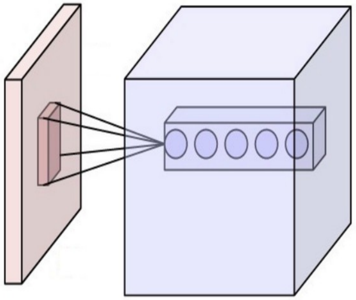
Filters: 100x(3x5x5)

Image : 3 (RGB) x 100 px x 100 px

Somewhat too many!

Convolution

Image : 3 (RGB) x 100 px x 100 px



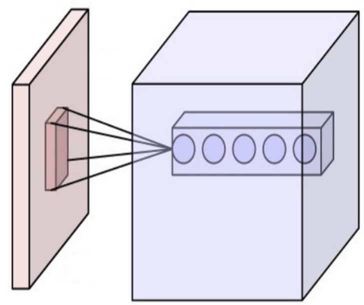
Filters: 100x(3x5x5)

→ 100x96x96

~10^6

100x96x96 → pool 3x4 → ???

Convolution



100x96x96

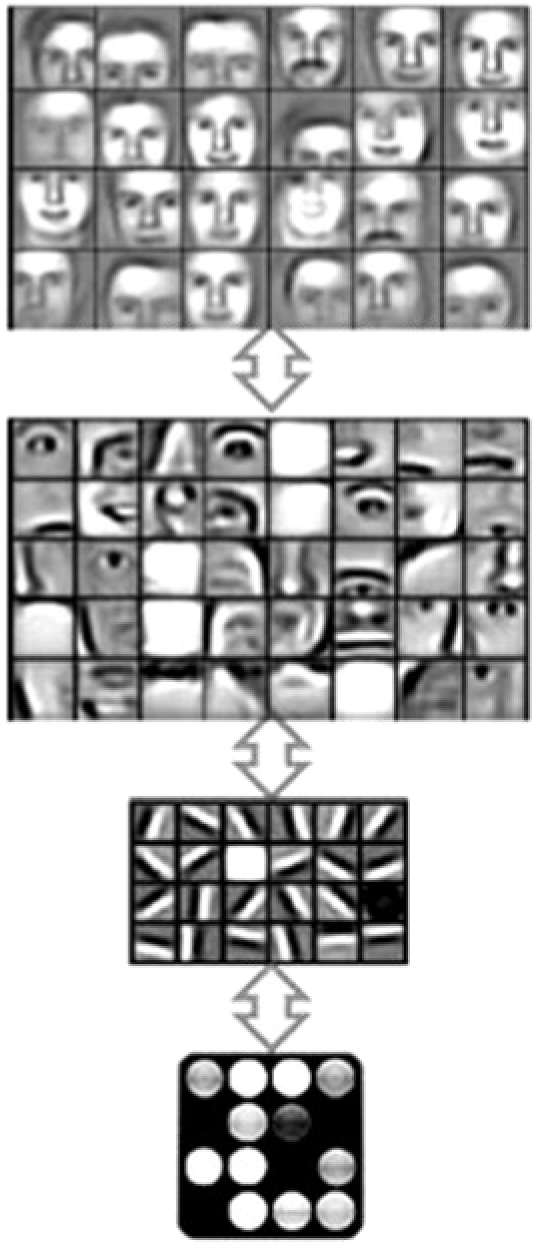
~10^6

Filters: 100x(3x5x5)

Image : 3 (RGB) x 100 px x 100 px

100x96x96 → pool 3x4 → 100x32x32

~10^5



Discrete Choices

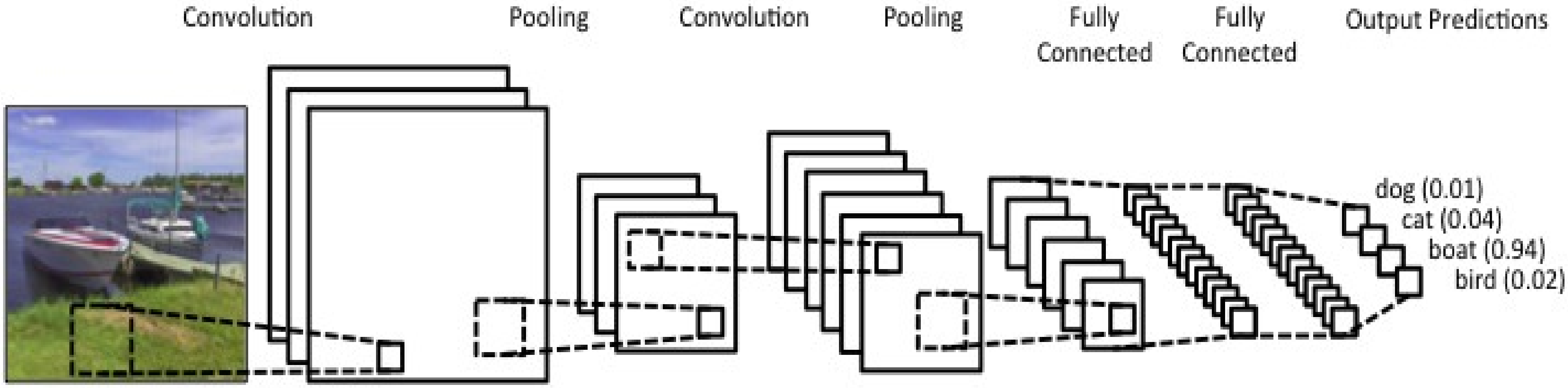
⋮

Layer 2 Features

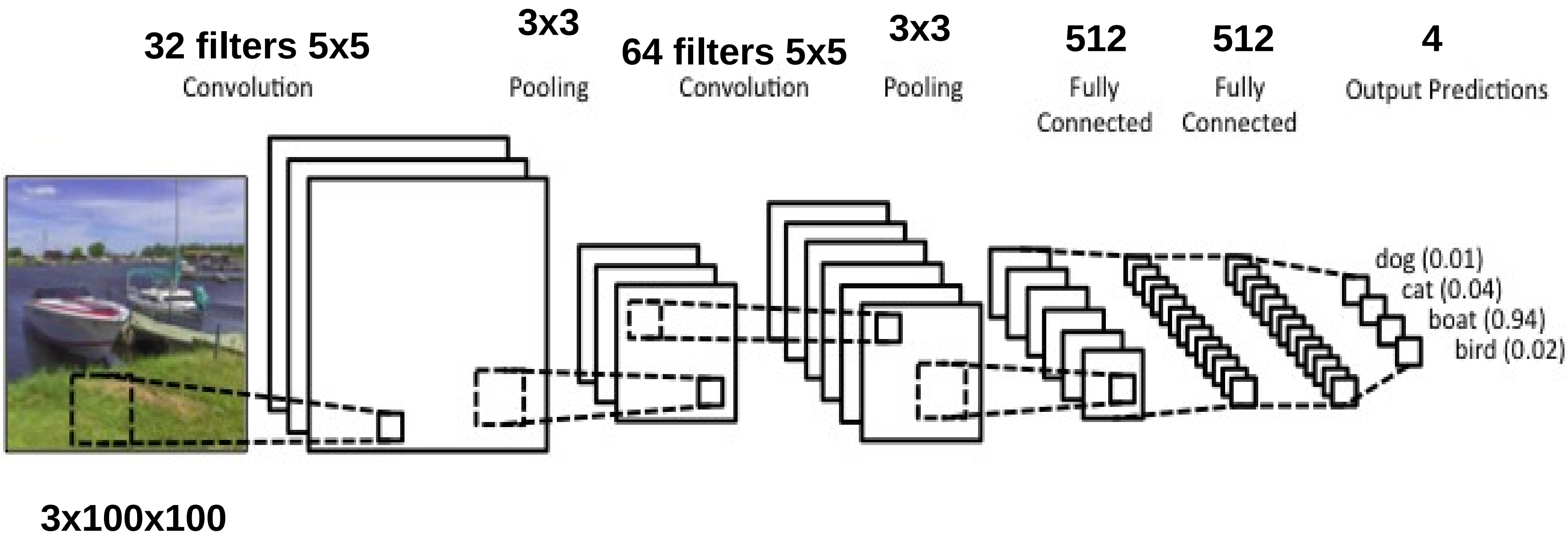
Layer 1 Features

Original Data

Convolutional NNs



Convolutional NNs

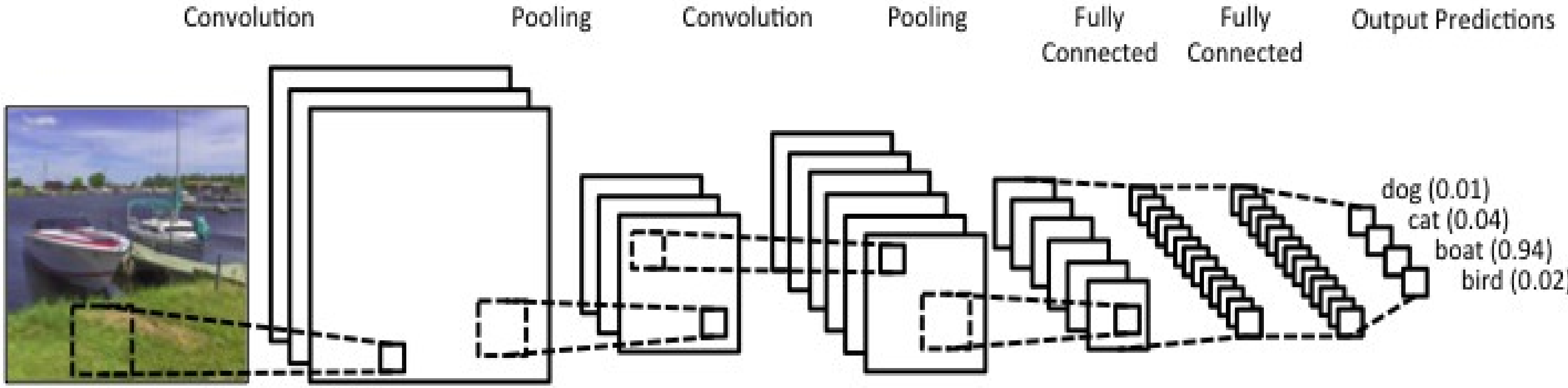


Quiz:

1) What is the output shape **after second pooling**

Convolutional NNs

32 filters 5x5 | 3x3 | 64 filters 5x5 | 3x3 | 512 | 512 | 4

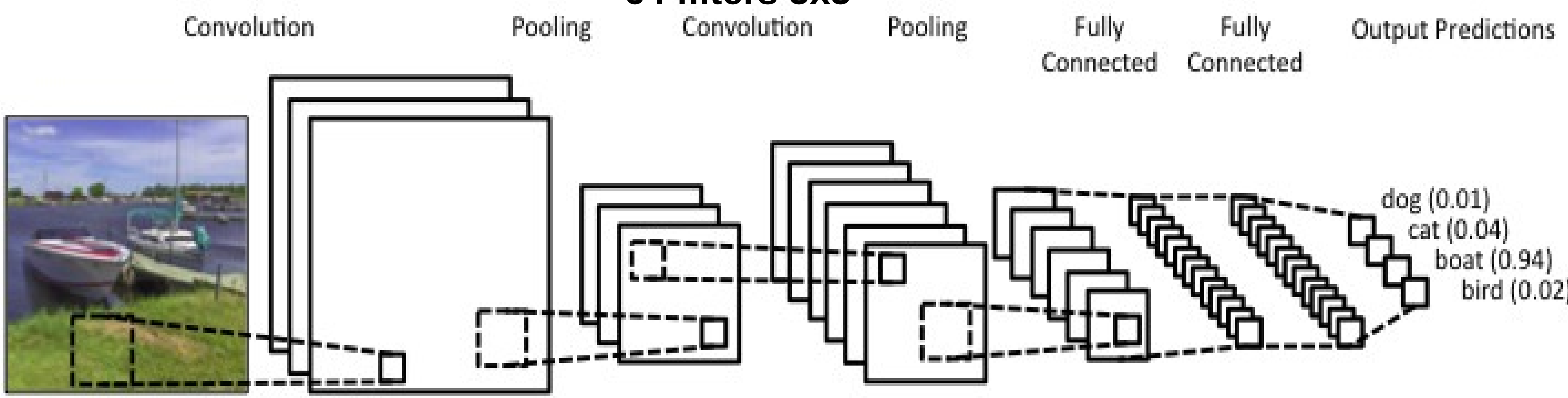


3x100x100

Quiz:
2) How many image pixels does **one cell** after **second convolution** depend on?

Convolutional NNs

32 filters 5x5 **3x3** **64 filters 5x5** **3x3** **512** **512** **4**



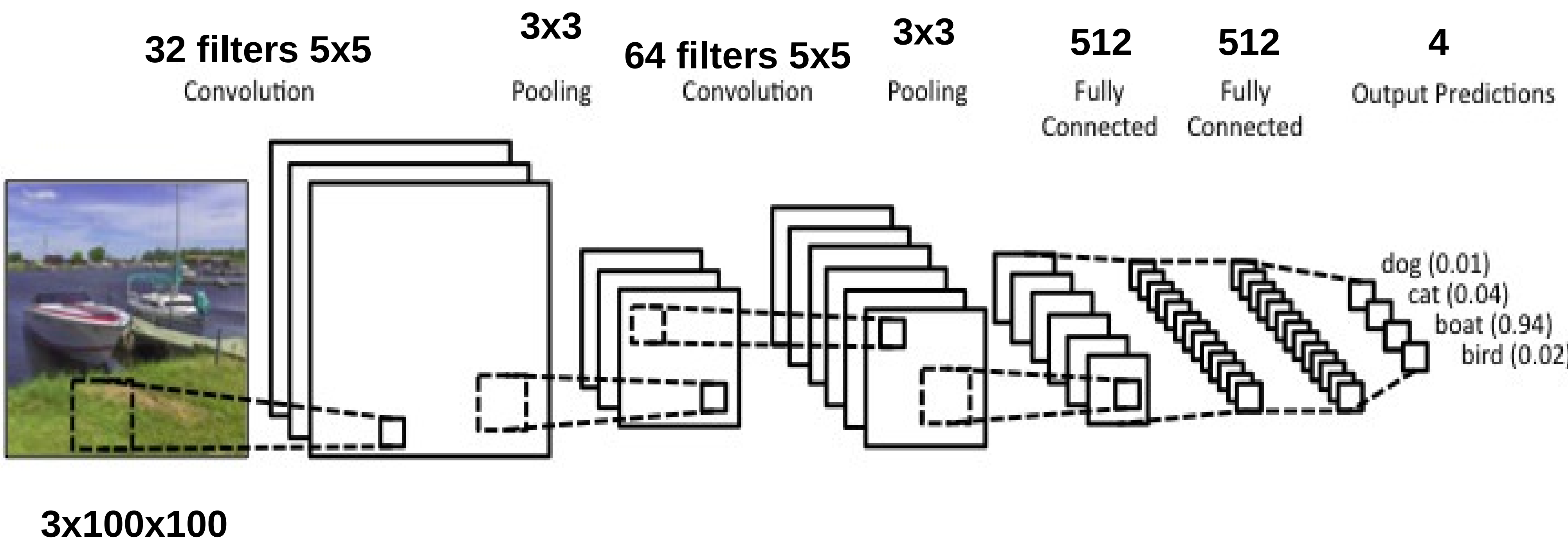
3x100x100

Quiz:

3) Which layer is hardest to compute?

4) Which layer has most independent parameters?

Convolutional NNs



Quiz:

3) Which layer is hardest to compute?: **first conv**

4) Which layer has most independent parameters? **first dense**

Problem with large networks

What you sign for if you stack 1000 layers:

- MemoryError(0x...)
- Gradients can vanish
- Gradients can explode
- Activations can vanish
- Activations can explode
- Overfits like crazy

Problem with large networks

What you sign for if you stack 1000 layers:

- MemoryError(0x...)
- **Gradients can vanish**
- **Gradients can explode**
- **Activations can vanish**
- **Activations can explode**
- Overfits like crazy

Q: How do we fix these?

Problem with large networks

What you sign for if you stack 1000 layers:

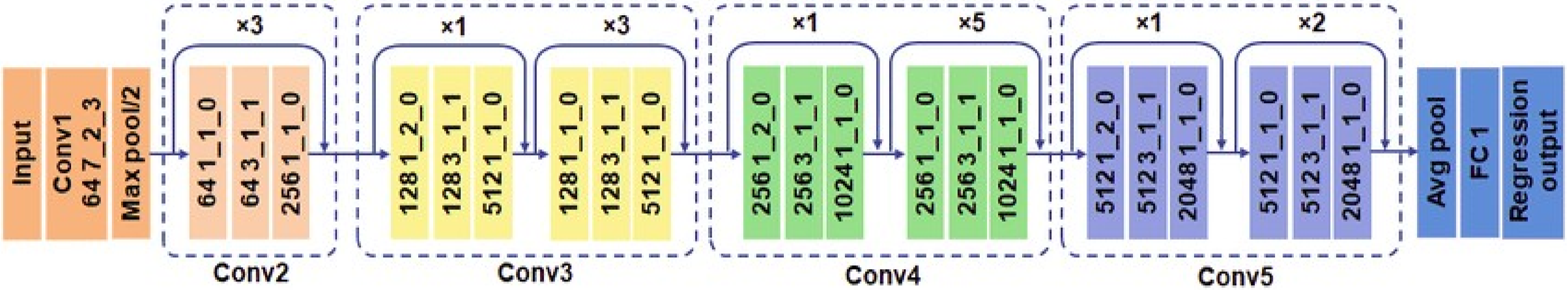
- MemoryError(0x...)
- **Gradients can vanish**
- **Gradients can explode**
- **Activations can vanish**
- **Activations can explode**
- Overfits like crazy

Residual connections

Batch normalization
or similar

ResNet & DenseNet

ResNet: add up activations



https://arxiv.org/abs/1512.03385

DenseNet: concatenate activations

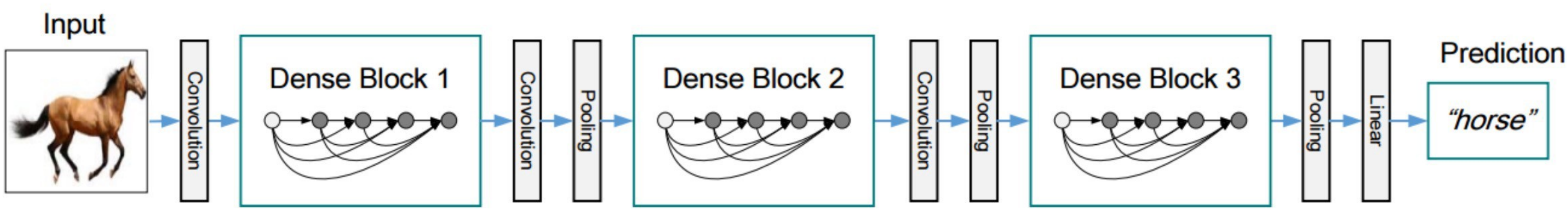
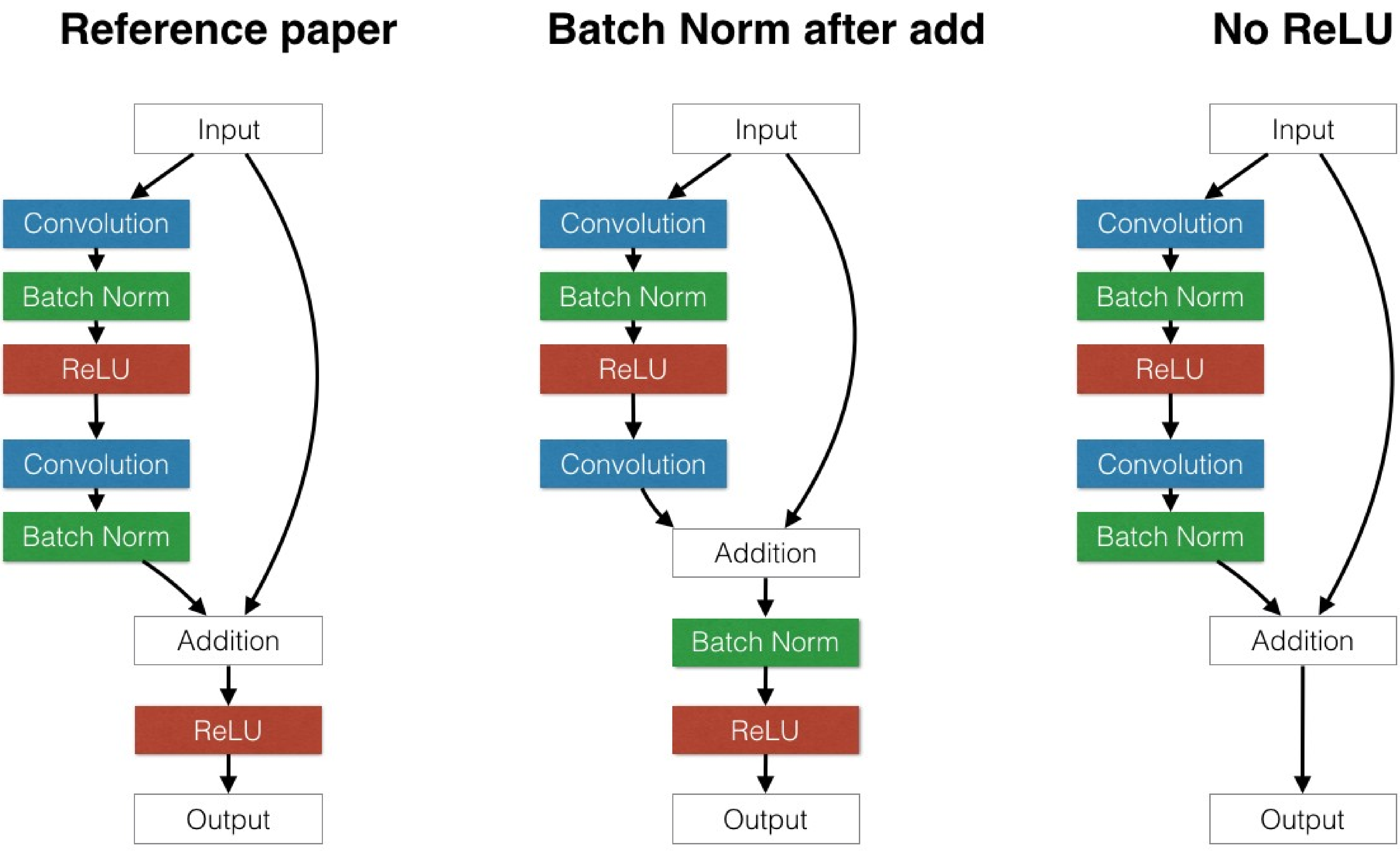


Figure 2. A deep DenseNet with three dense blocks. The layers between two adjacent blocks are referred to as transition layers and change feature map sizes via convolution and pooling.

https://arxiv.org/abs/1608.06993

ResNet architectures



Problem with large networks

What you sign for if you stack 1000 layers:

- MemoryError(0x...)
- **Gradients can vanish**
- **Gradients can explode**

Residual connections

- **Activations can vanish**
- **Activations can explode**

Batch normalization or similar

- Overfits like crazy

Data augmentation

- Idea: we can get N times more data by tweaking images.
- If you rotate cat image by 15°, it's still a cat

- Rotate, crop, zoom, flip horizontally, add noize, etc.
- Sound data: add background noizes

Modern stuff

- Architectures:

  EfficientNetV2 https://arxiv.org/abs/2104.00298
  CNN for 2020 https://arxiv.org/abs/2201.03545

- Interpretability

  https://distill.pub/2018/building-blocks

Other CV applications

Real computer vision starts when
image classification is no longer enough.

Bounding box regression

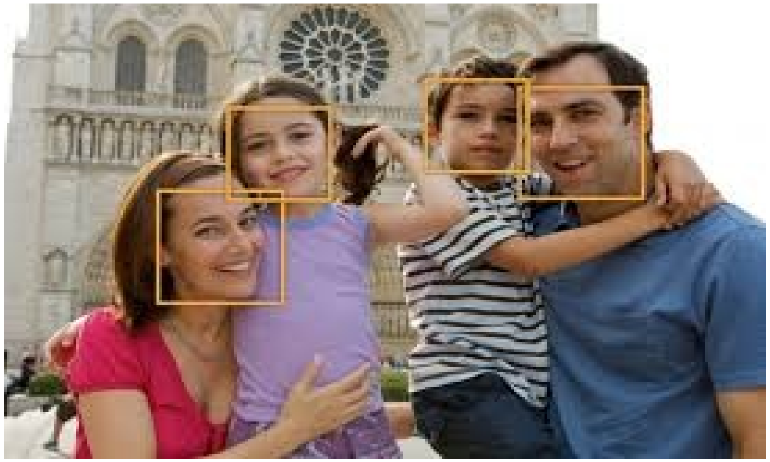
Predict object bounding box

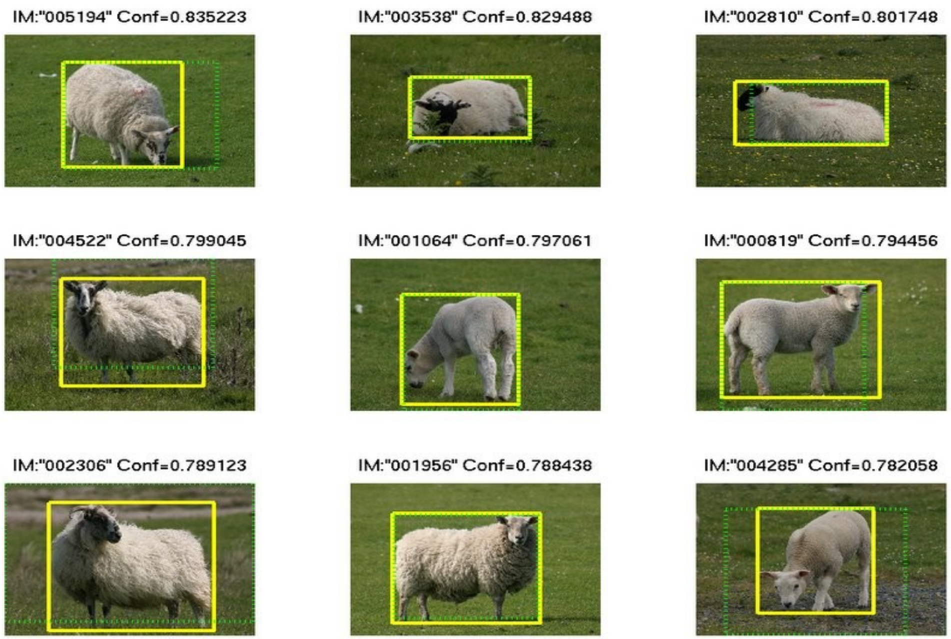
(x0,y0,w,h)

or several bounding boxes for multiple objects.

Applications examples:

- Face detection @ cameras
- Surveillance cameras
- Self-driving cars



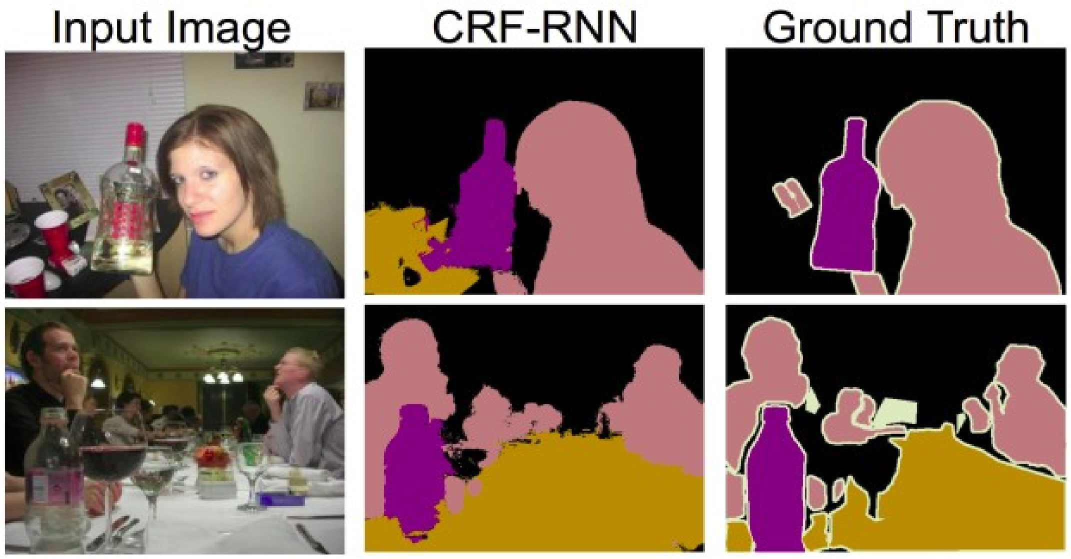


Segmentation

Predict class for each pixel

(fully-convolutional networks)

Applications examples:
- Moar surveillance
- Brain scan labeling
- Map labeling



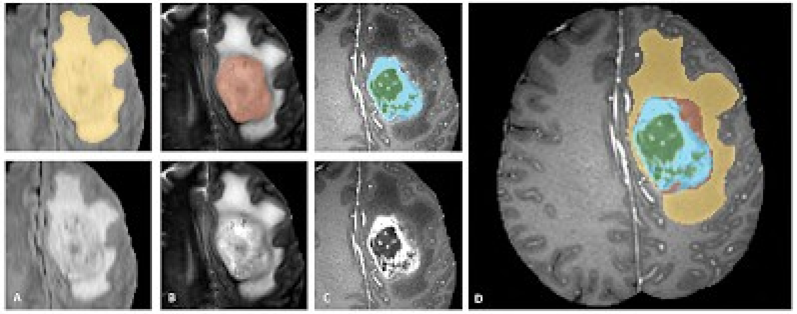
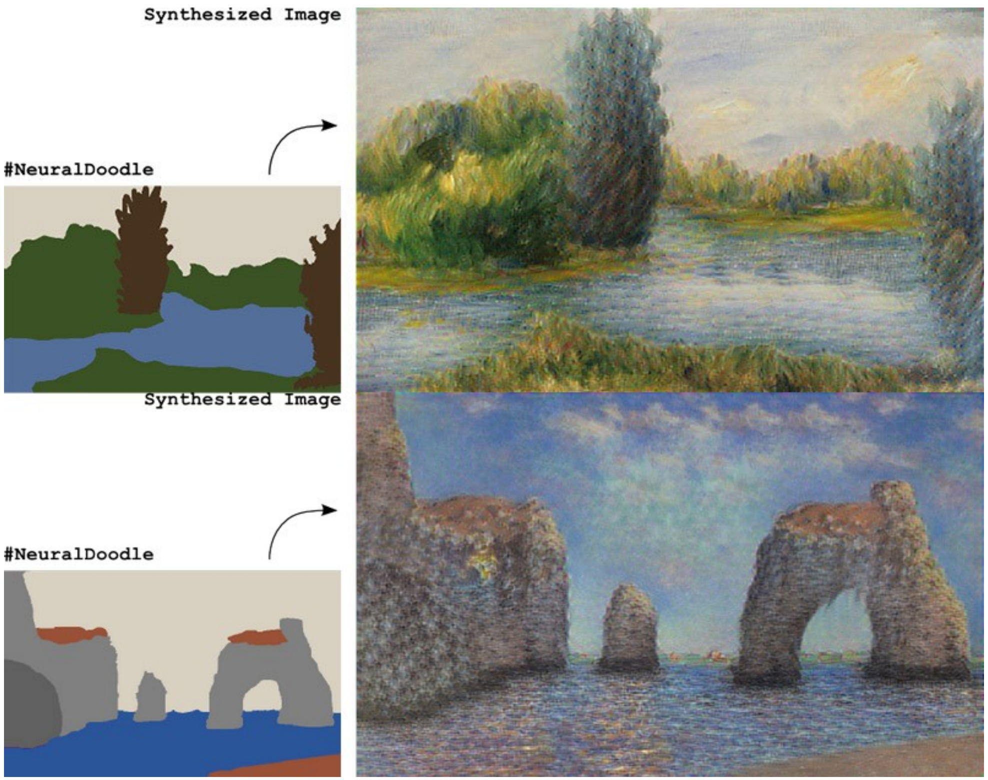


Image generation/transformation

- **Generation:** Given a set of reference images, learn to generate new images, resembling those you were given.

- **Transformation:** Given a set of reference images, learn to convert other images into ones resembling the reference set.



Neural Doodle
(D. Ulyanov et al.)

Image tagging
Image captioning
Image retrieval
Image encoding
Image morphing
Image encoding
Image upscaling
Object tracking on video
Video processing
Video interpolation

Fine-tuning
Adversarial Networks
Variational Autoencoders
Knowledge transfer
Domain adaptation
Online learning
Explaining predictions
Soft targets
Scene reconstruction
3D object retrieval
Classifier optimization

Nuff

Let's train some CNNs!