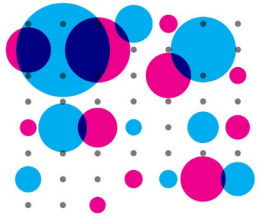


CGnal

business innovation through algorithms

# Convolutional Neural Networks

CGnal s.r.l. – Corso Venezia 43 - Milano

23 novembre 2021| Milano

DAY 1

## Introduction

- Brief overview of Machine Learning (Supervised, Unsupervised)
- Introduction to Graph, Graph Theory and main metrics for characterizing graphs

DAY 2

## Graph Machine Learning

- Community detection on Graphs
- Supervised Machine Learning on Graphs

DAY 3

## Explainability & Interpretability

- Introduction to explainability problem
- LIME & SHAP

DAY 4

## Simple Neural Networks

- Introduction to Neural Networks, TensorFlow and Computational Graphs
- Implementation and training of simple Neural Networks

DAY 5

## Advanced Neural Networks

- Convolutional Neural Networks and Recurrent Neural Networks
- Advanced Topics

# Computer vision

A gray scale image is a **matrix of pixels** [H x W]

Pixel = **pic**ture **el**ements

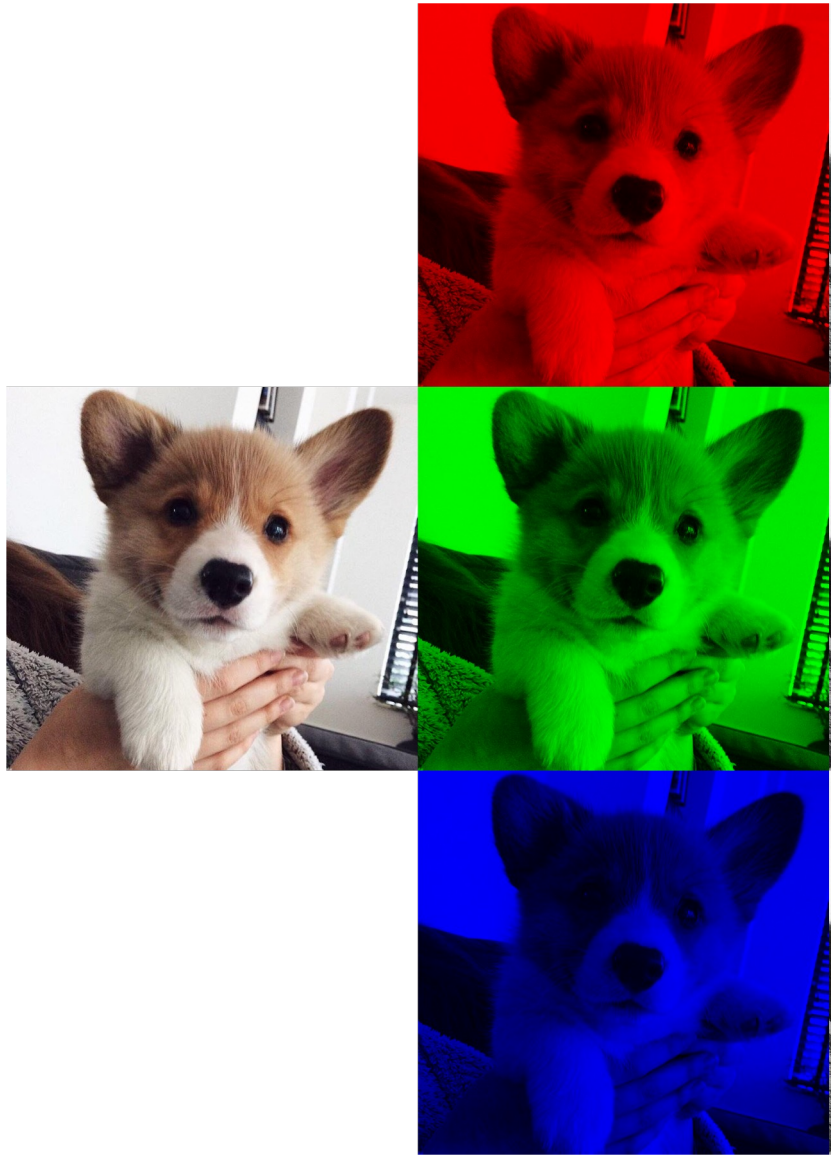
Each pixel is an integer value [0, 255] representing the brightness

# Computer vision

A gray scale image is a **matrix of pixels** [H x W]

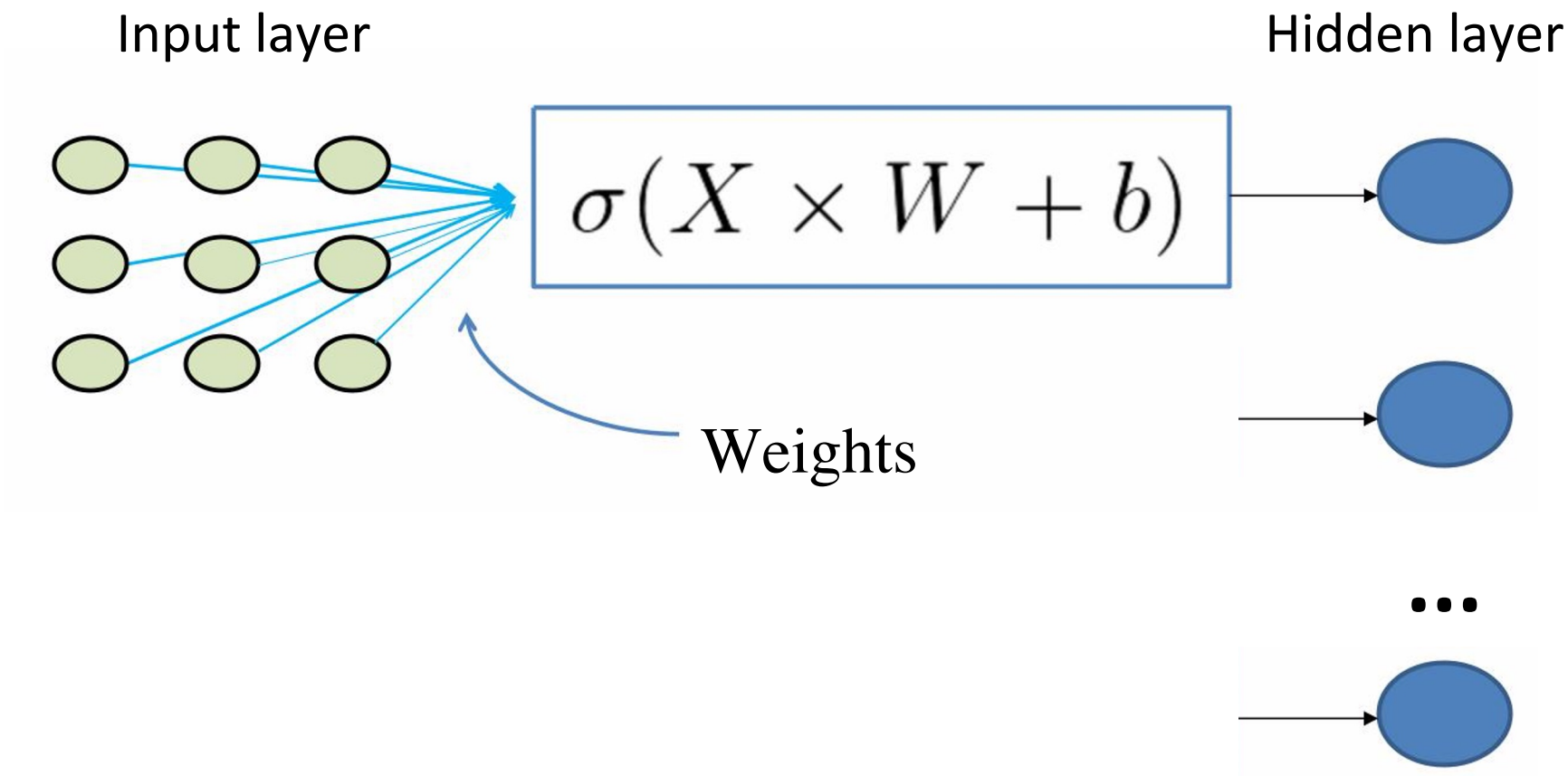
Pixel = **pic**ture **el**ements

Each pixel is an integer value [0, 255] representing the brightness

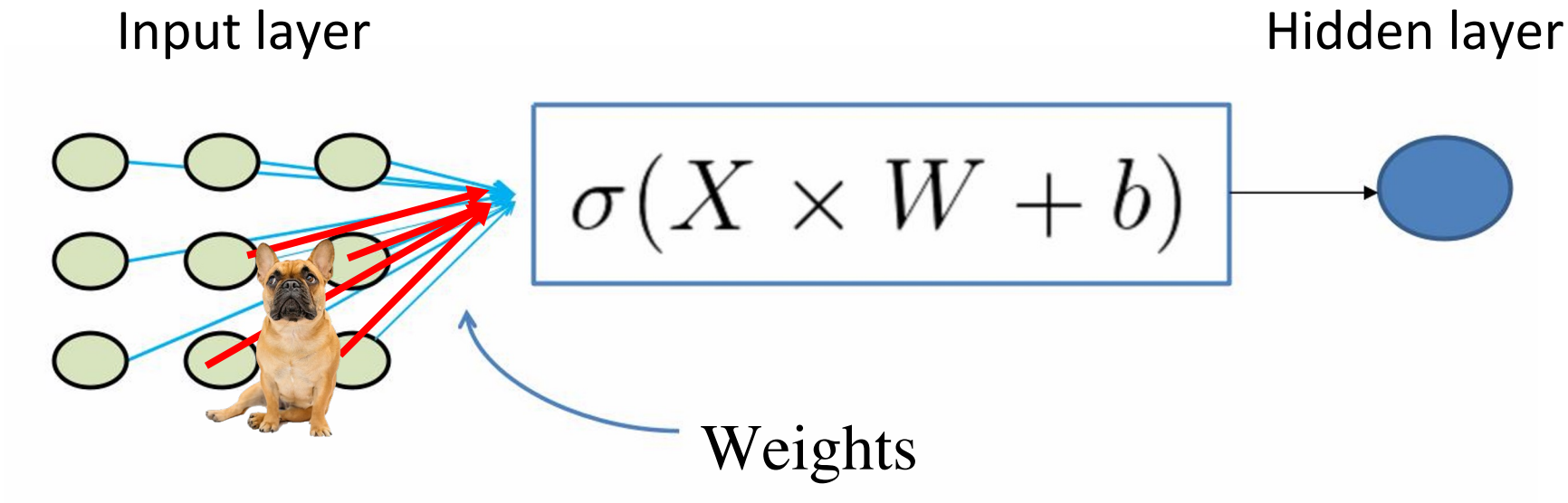


An RGB image is a 3D array [H x W x 3]

# Image recognition and deep learning

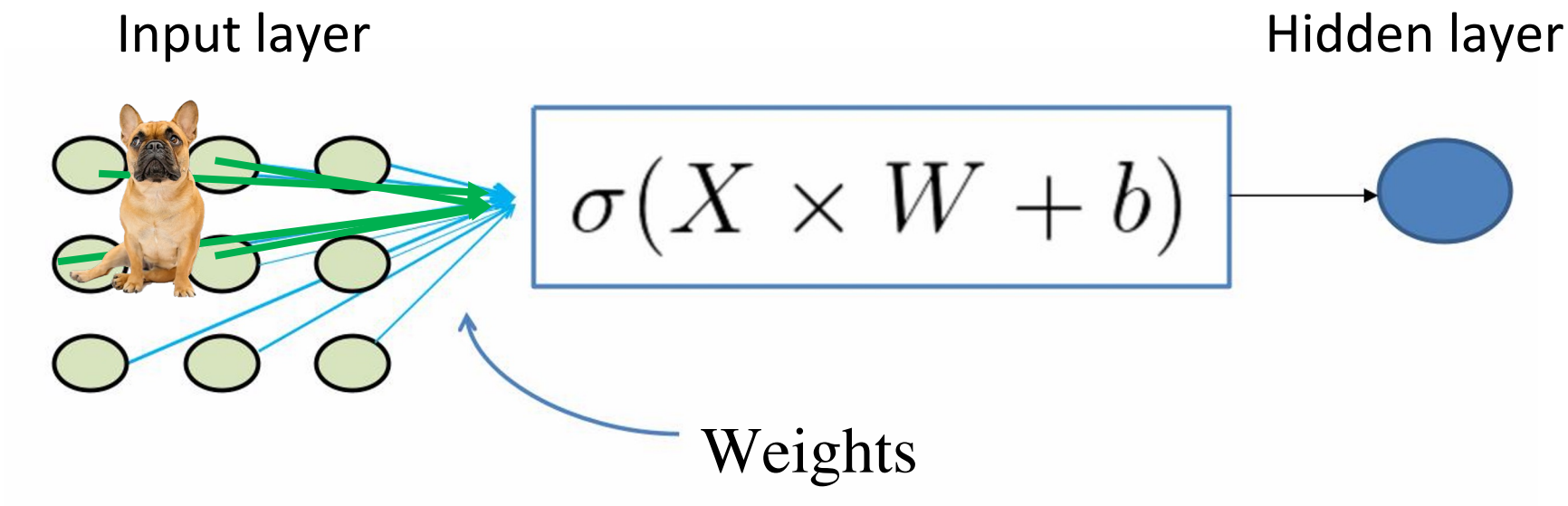


# Image recognition and deep learning



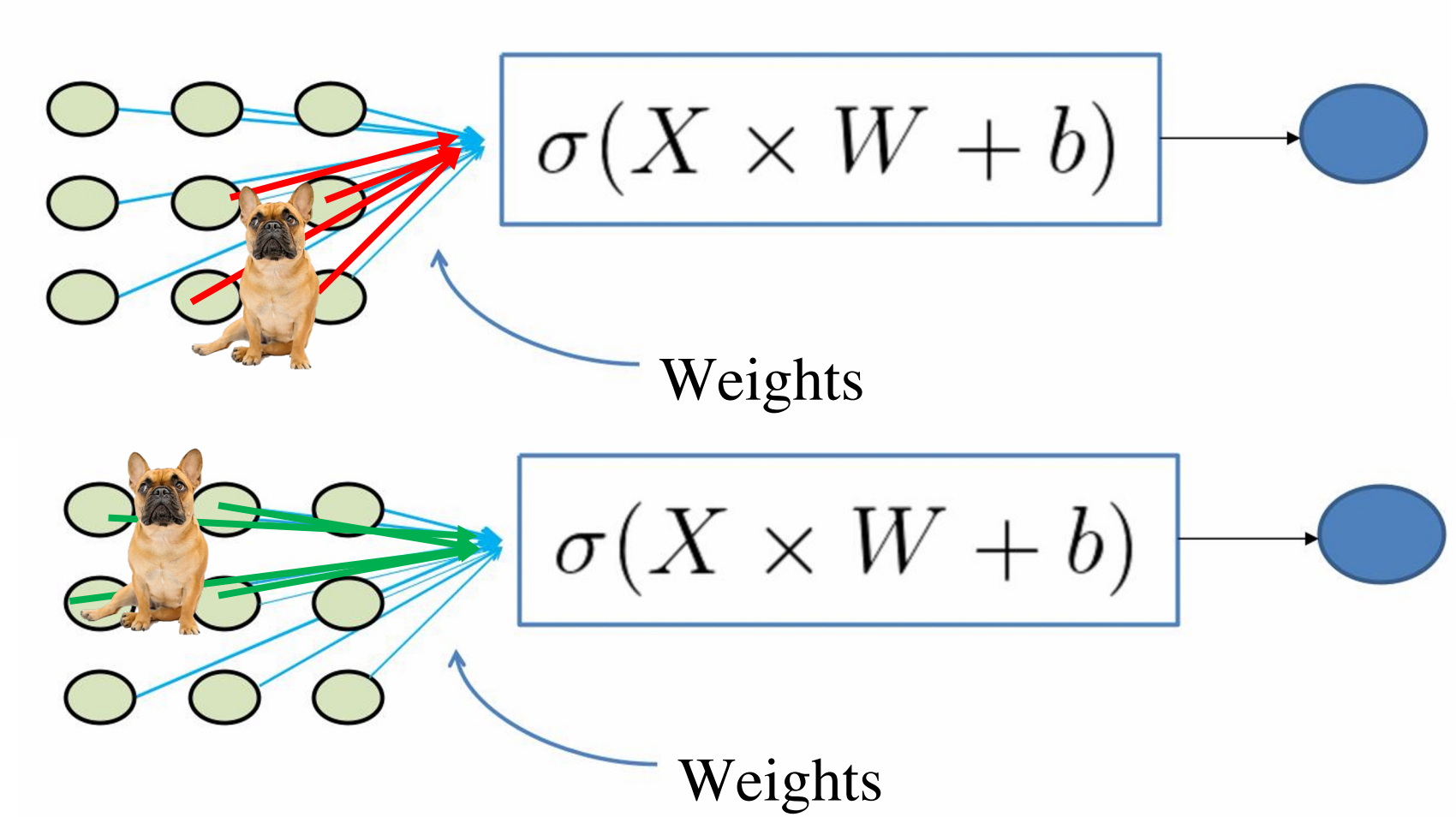

On this object, you will train **red** weights to react on the dog

# Image recognition and deep learning




On this object, you will train **green** weights to react on the dog

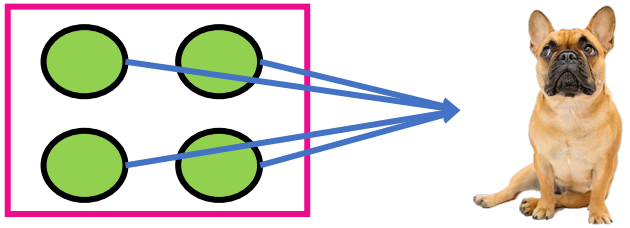
# Image recognition and deep learning




**The network will have to learn these two cases separately: very inefficient!**

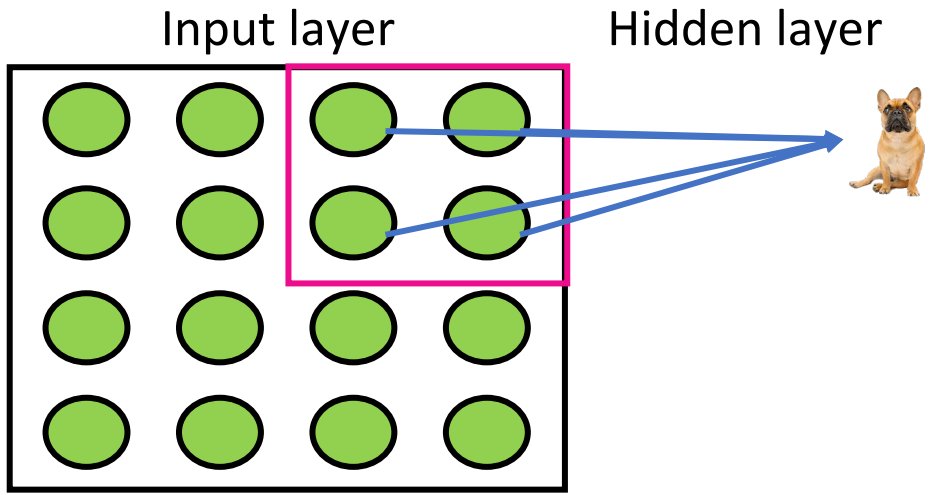
# Convolutional Neural Networks

Main idea: let ’s encode "dog" by a weight tensor that we can shift across the image.



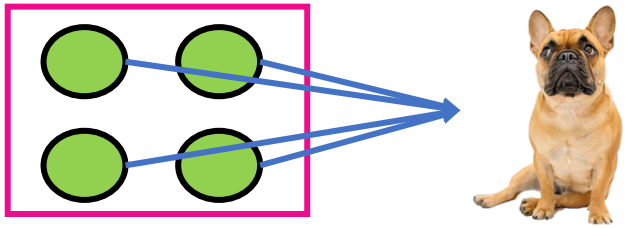


*Kernel* = "portable dog detector"



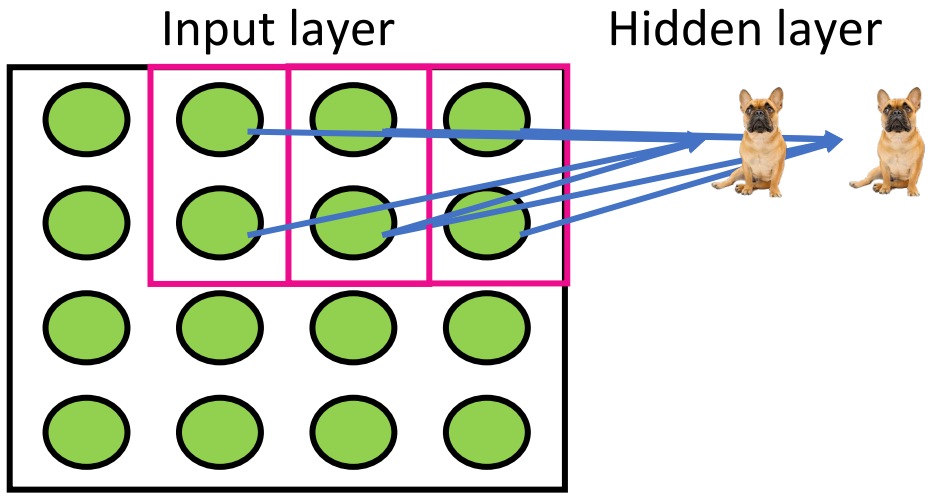
# Convolutional Neural Networks

Main idea: let ’s encode "dog" by a weight tensor that we can shift across the image.



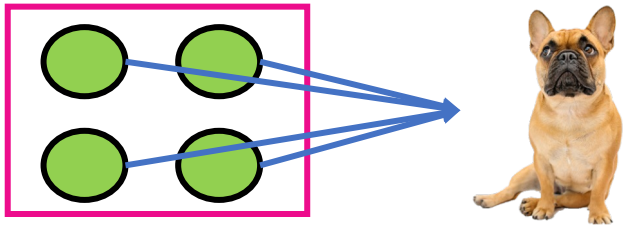


*Kernel* = "portable dog detector"



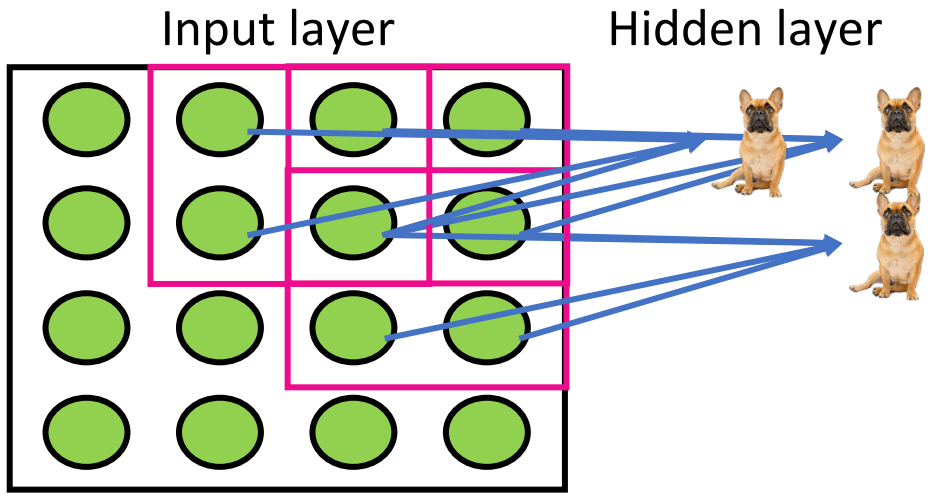
# Convolutional Neural Networks

Main idea: let ’s encode "dog" by a weight tensor that we can shift across the image.



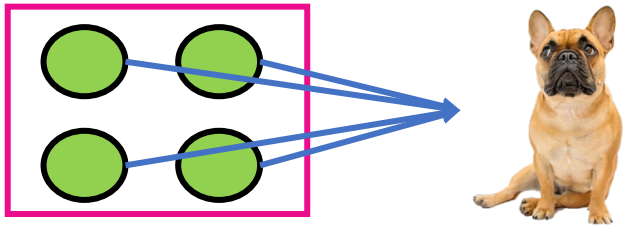


*Kernel* = "portable dog detector"



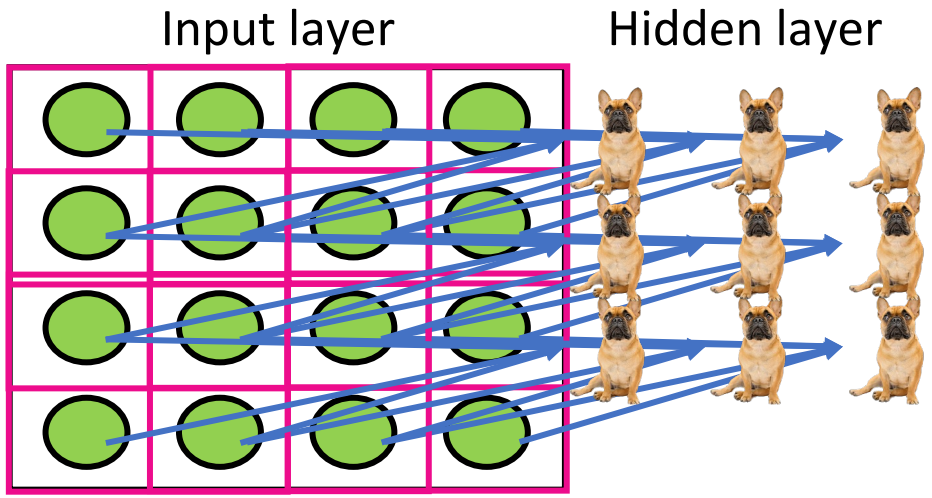
# Convolutional Neural Networks

Main idea: let ’s encode "dog" by a weight tensor that we can shift across the image.





*Kernel* = "portable dog detector"



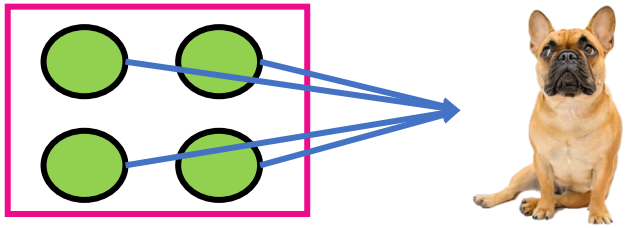

**Input layer: 4x4 = 16 pixels**

**Conv layer = 3x3 = 9 activations**

Activations represent "dogness" of the 2x2 window, i.e. probability that a dog is in that area of the picture

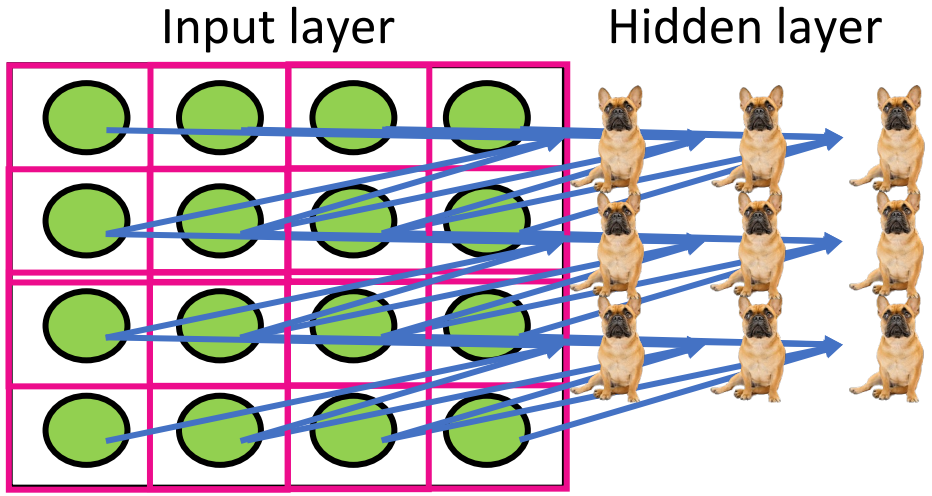
# Convolutional Neural Networks

Main idea: let ’s encode "dog" by a weight tensor that we can shift across the image.





*Kernel* = "portable dog detector"

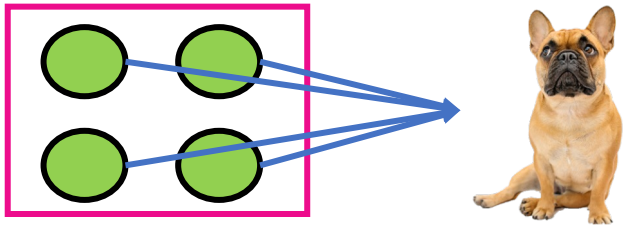



**Input layer: 4x4 = 16 pixels**
**Conv layer = 3x3 = 9 activations**

**How many trainable weights?**

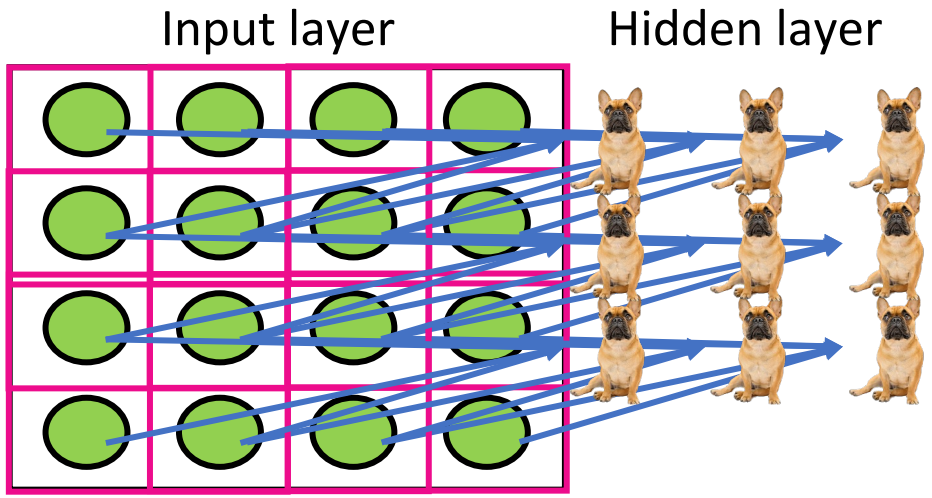
# Convolutional Neural Networks

Main idea: let ’s encode "dog" by a weight tensor that we can shift across the image.





*Kernel* = "portable dog detector"



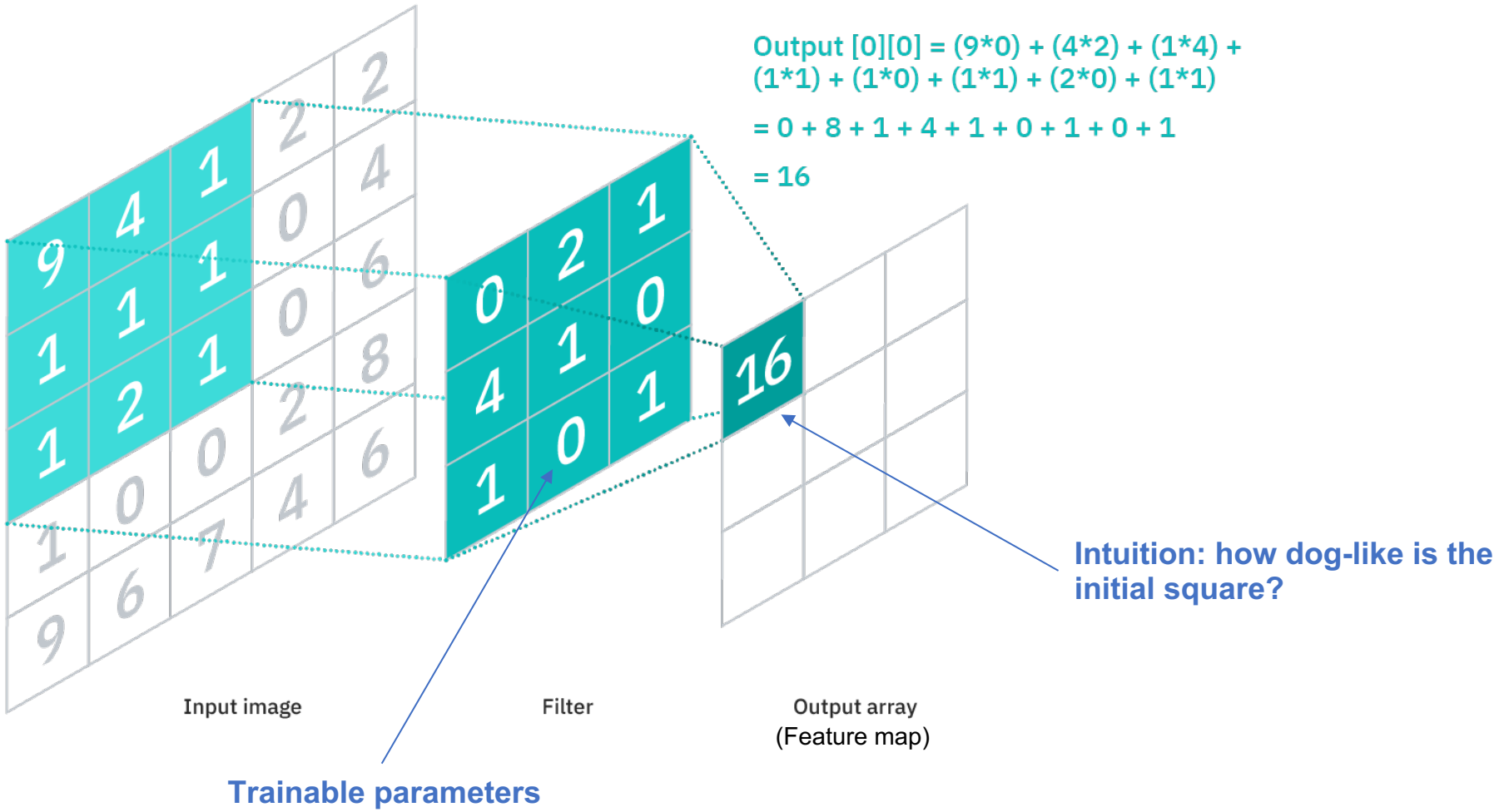

**Input layer: 4x4 = 16 pixels**
**Conv layer = 3x3 = 9 activations**

**How many trainable weights?**

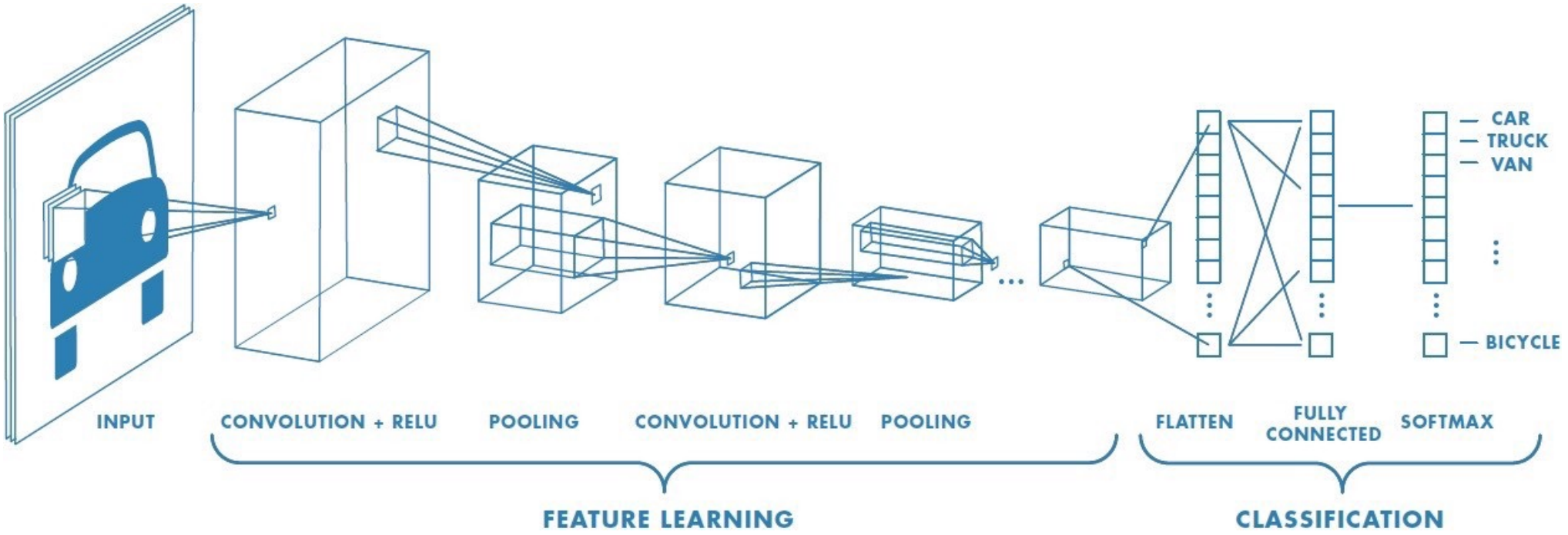
**2x2 kernel = 4 trainable weights**

# Convolution

For each portion of the image:



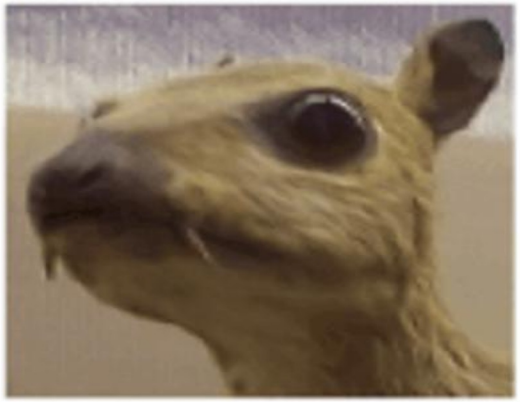
# Multiple convolutional layers




Different convolutional layers extract different ‘abstract’ features, with **subsequent level of complexity**

# Convolution

Input image



Convolution Kernel

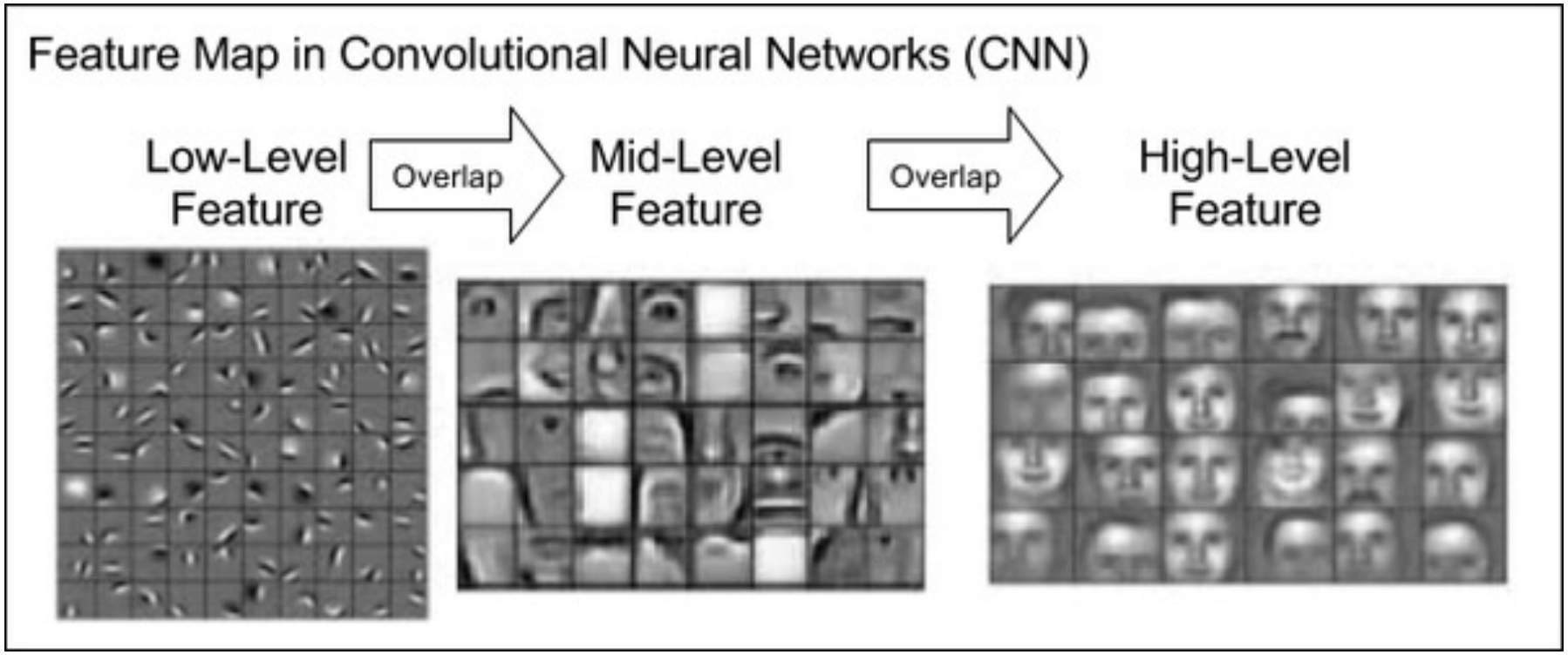
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Feature map

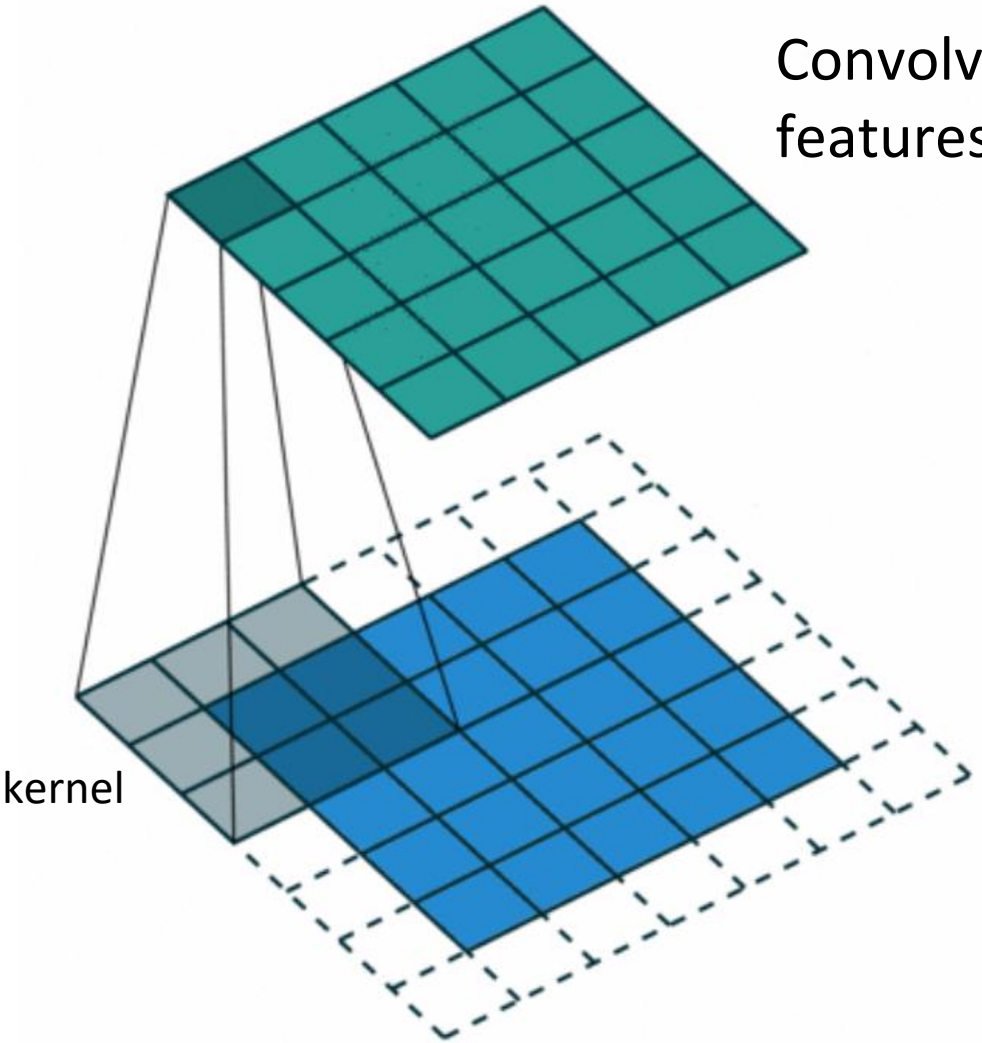
Intuition: how edge-like is this square?

Demo: https://demonstration-storage.education-team.ru/convolution/index.html

# From low to high level features

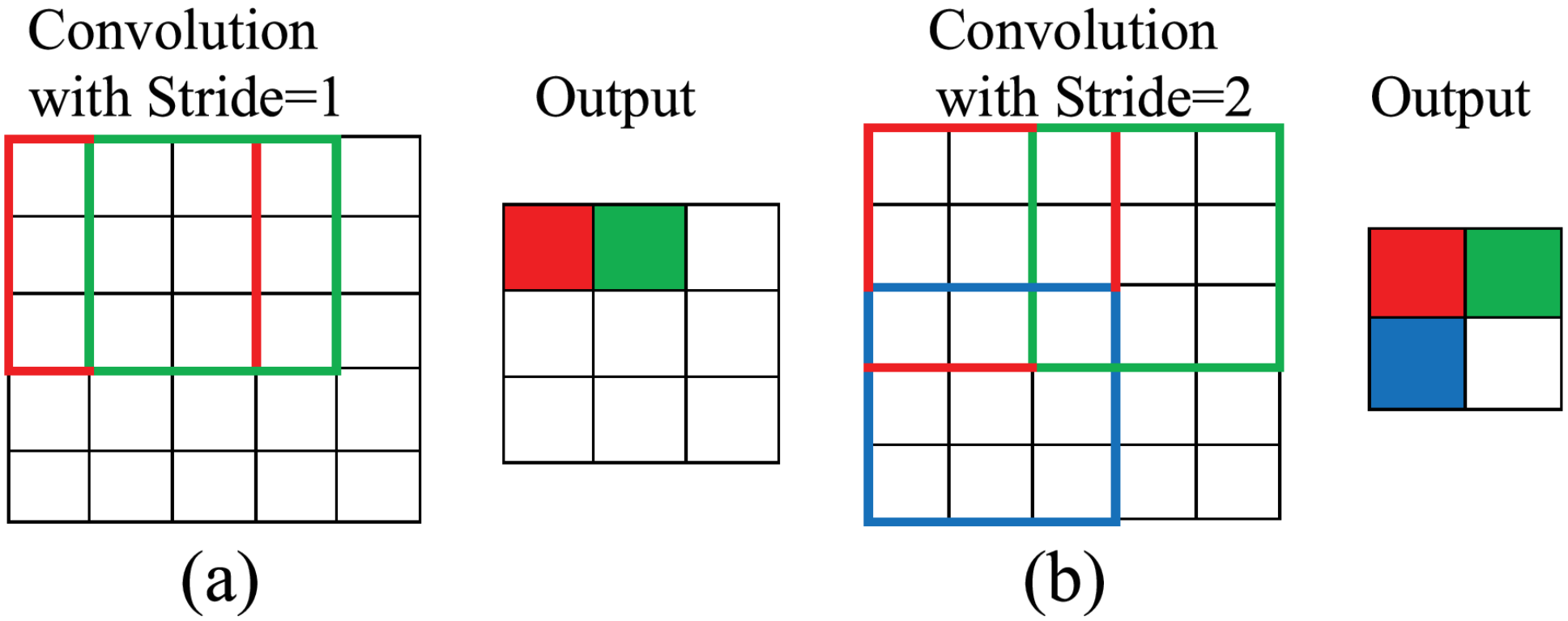


# Padding




- Pad the edges with extra, “fake” pixels
  - usually of value 0, hence the oft-used term “**zero padding**”
- The kernel when sliding can allow the original edge pixels to be at its centre, while extending into the fake pixels beyond the edge, producing an **output the same size as the input**

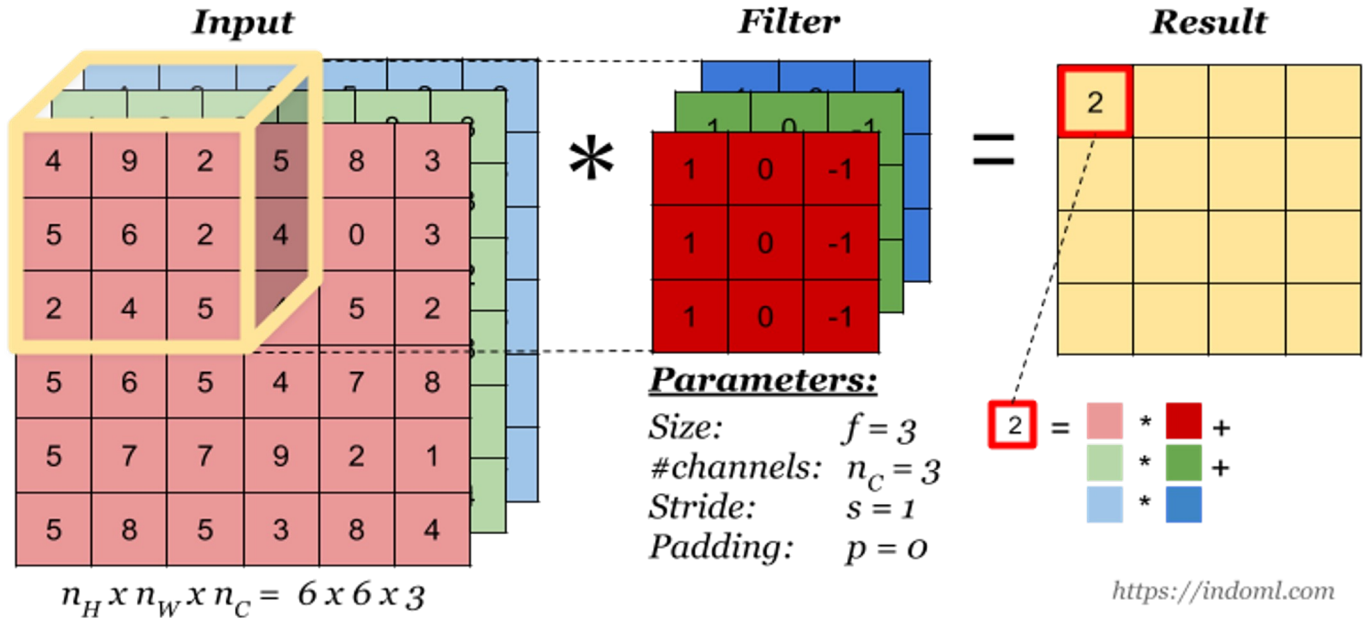
# Striding




- Skip some of the slide locations of the kernel
- **Reduce the size of the output** (and the number of parameters to be learned)

# Number of channels/kernels

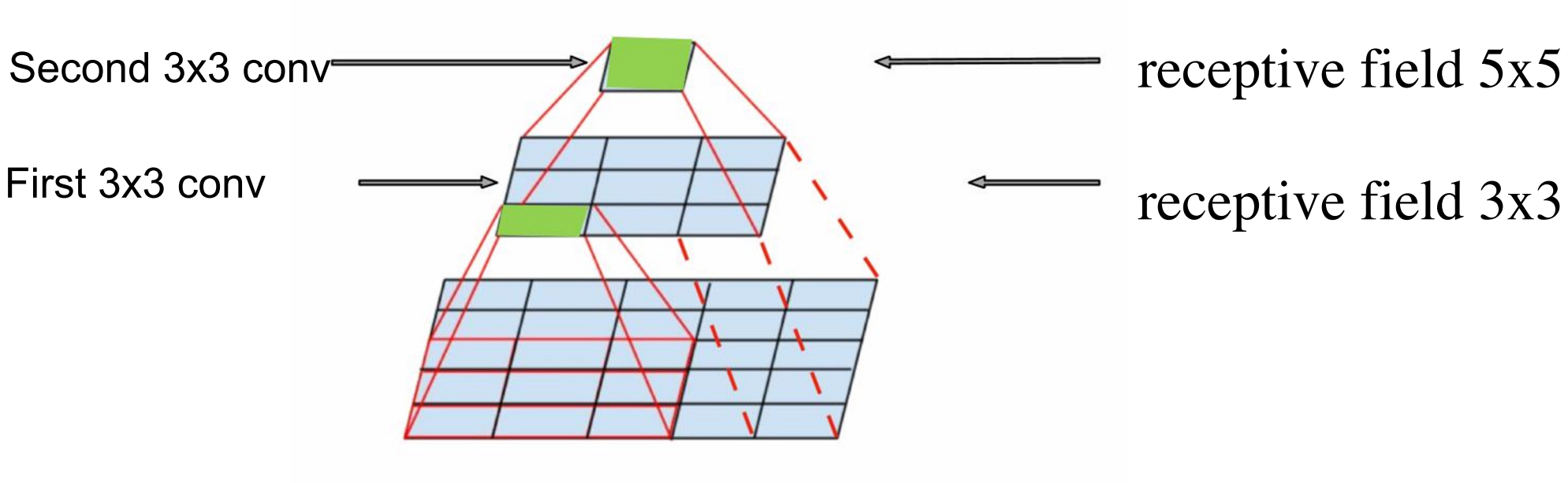
- A conv layer may have **multiple input channels**, e.g. 3 channels for RGB
  - one 2D kernel per channel with same dimensions and independent trainable weights, i.e. kernel is a 3D matrix (H x W x C)
  - the output of each channel is typically aggregated, e.g. summed



# Number of channels/kernels

- A conv layer may have **multiple input channels**, e.g. 3 channels for RGB
  - one 2D kernel per channel with same dimensions and independent trainable weights, i.e. kernel is a 3D matrix (H x W x C)
  - the output of each channel is typically aggregated, e.g. summed
- A single conv layer may learn **arbitrary number of kernels**
  - each output channel corresponds to a kernel
  - the output channels of a conv layer has a different meaning than color channels, generally not (easily) interpretable

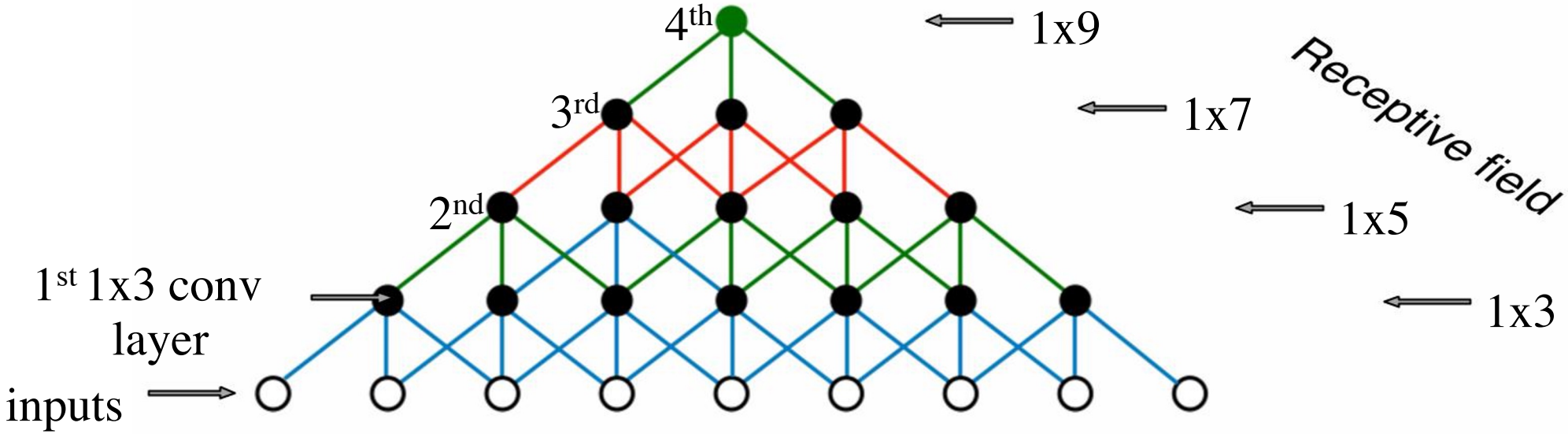
# Receptive field




We can recognize larger objects by stacking several small convolutions!

# Receptive field

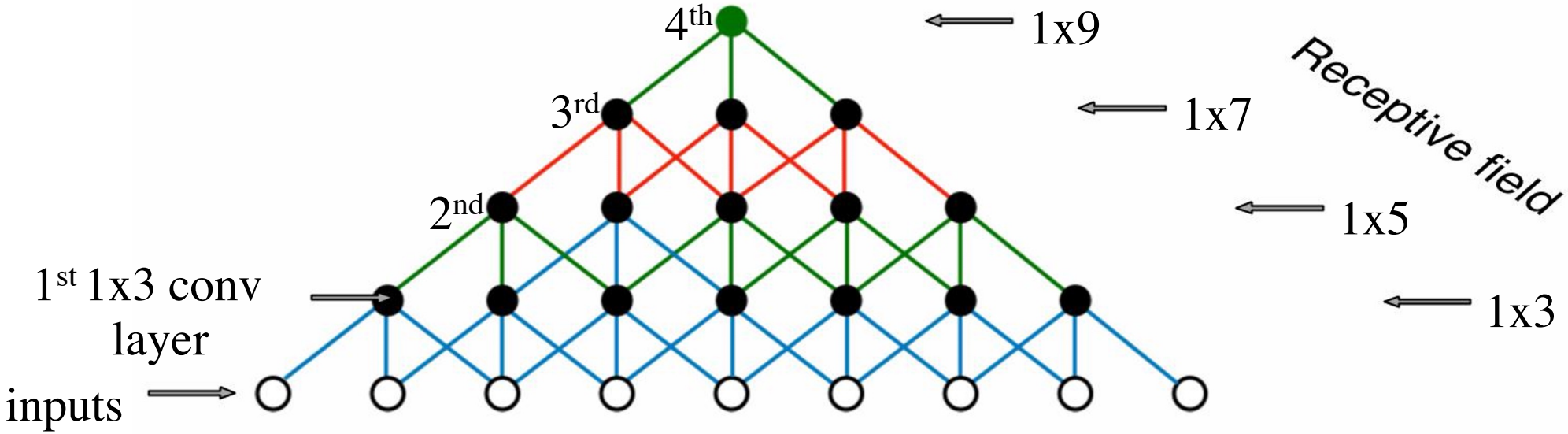
1-dimensional example




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

# Receptive field

1-dimensional example

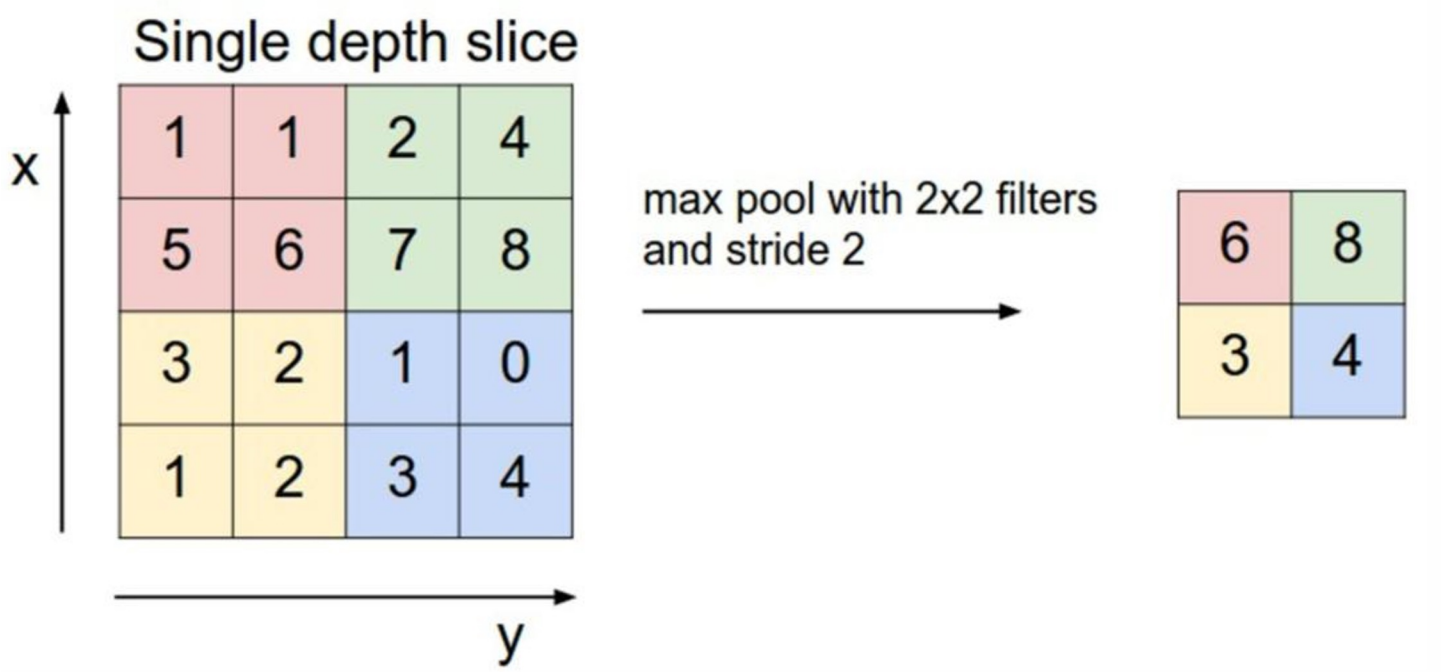



Q: how many 3x3 convolutions we should use to recognize a 100x100px dog?

**About 50!**

https://distill.pub/2019/computing-receptive-fields/

# Pooling




Intuition: highest "dogness" in the 2x2 area

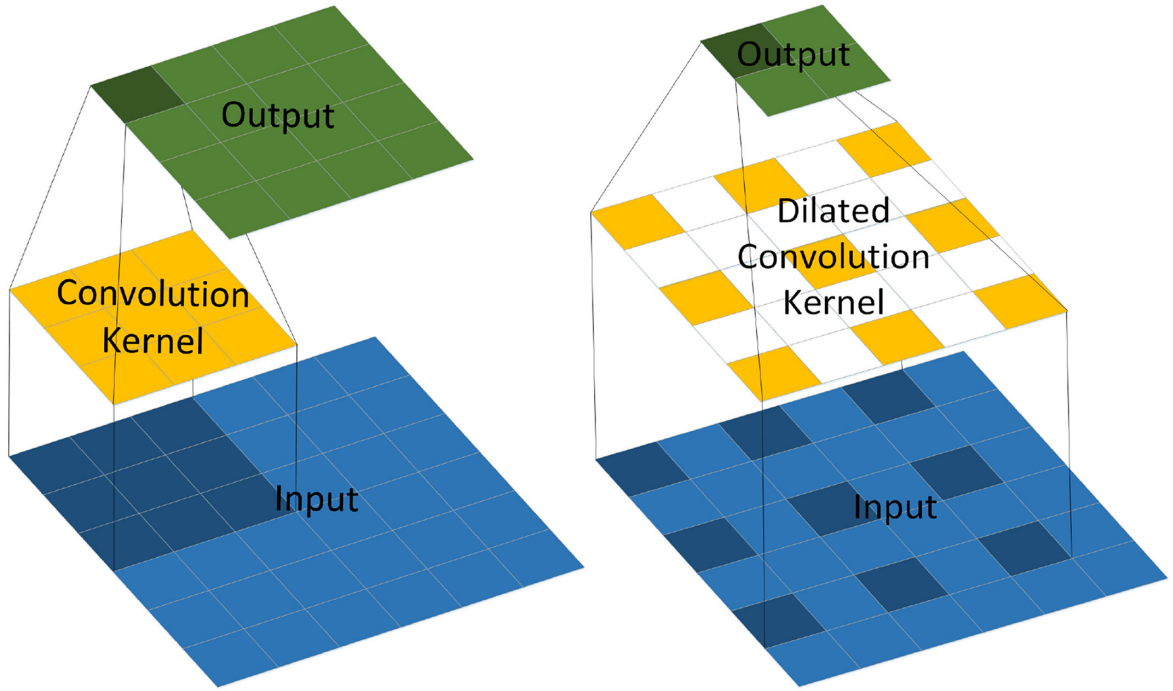
Increase receptive field faster

Reduce layer size

Make NN less sensitive image shifts

Popular types: max, mean

# Dilation



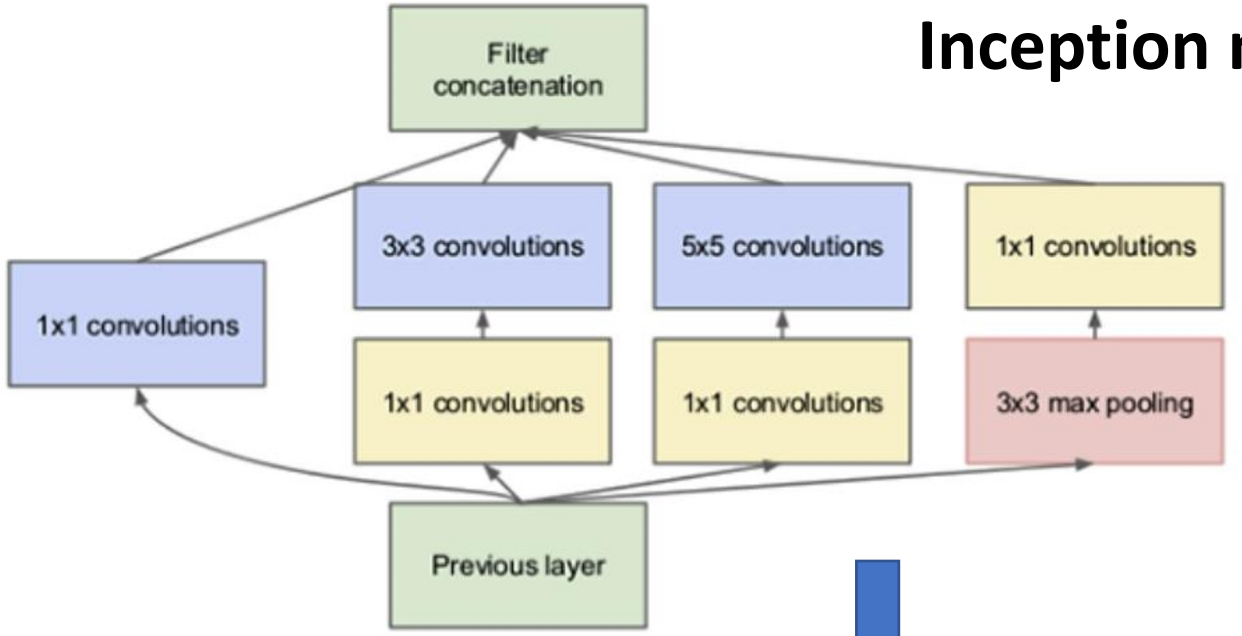

(A) Dilation = 1 (B) Dilation = 2

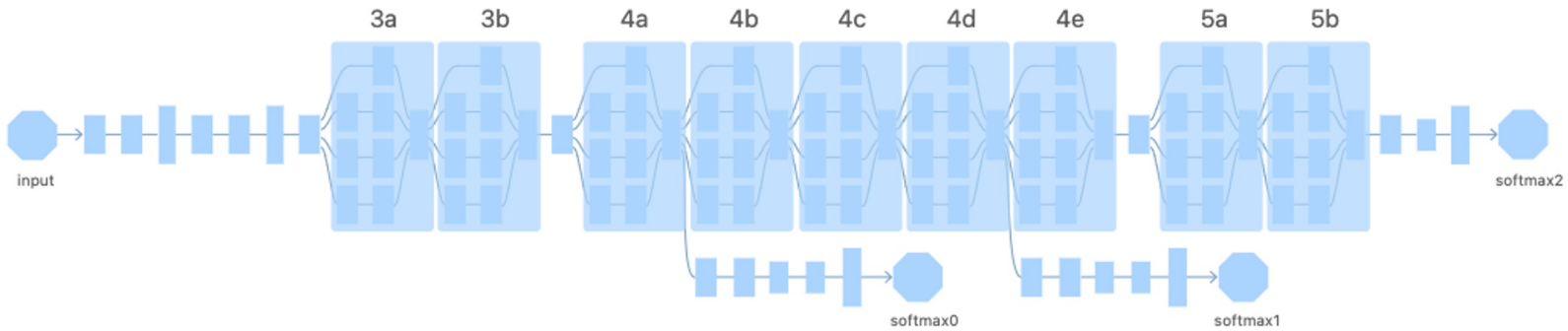
Dark points = pixels multiplied by the kernel

- Increased receptive field with same number of parameters
- Similar to reducing image resolution, but some fine-grained details are kept

# State-of-the-art computer vision: GoogleNet

## Inception module

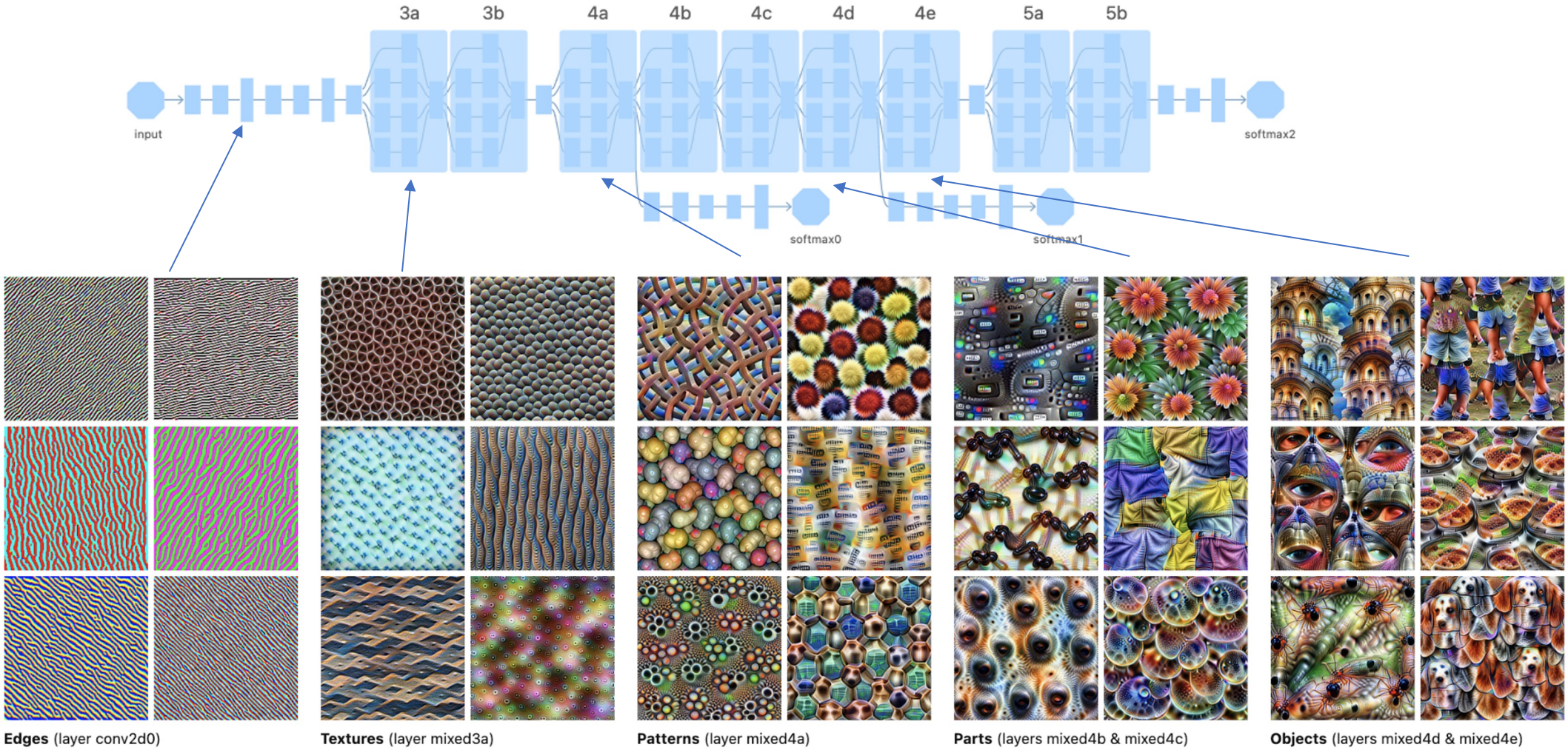



CGnal

# Computer vision model interpretability

https://distill.pub/2017/feature-visualization/



# Summary

- Convolutional Neural Networks are **engineered networks** based on intuitive ideas and experience with image-related tasks
- A convolutional layer is defined by a its *kernels* (size and number), *stride*, *padding*, *dilation*
- **Pooling** is used to reduce the previous layer size and make the NN less sensitive to small image shifts
- State-of-the-art computer vision models are **highly engineered networks**, e.g. *GoogleNet*

# Hands on

# Exercise

# **Convolutional Neural Networks**