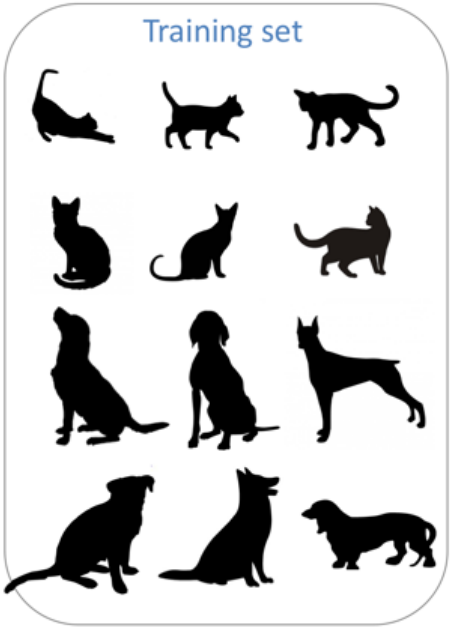
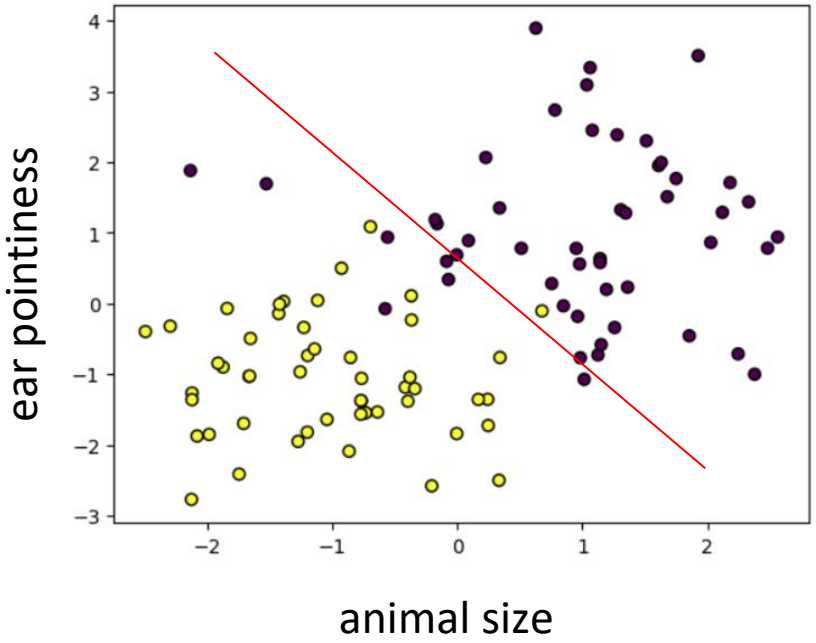
# Tutorial 3: Supervised learning and feedforward neural networks

# Plan

- Perceptron.
- Multi-layer Perceptron.
- Convolutional Neural Network and digit recognition
- Analogy between convolutional networks and visual cortex

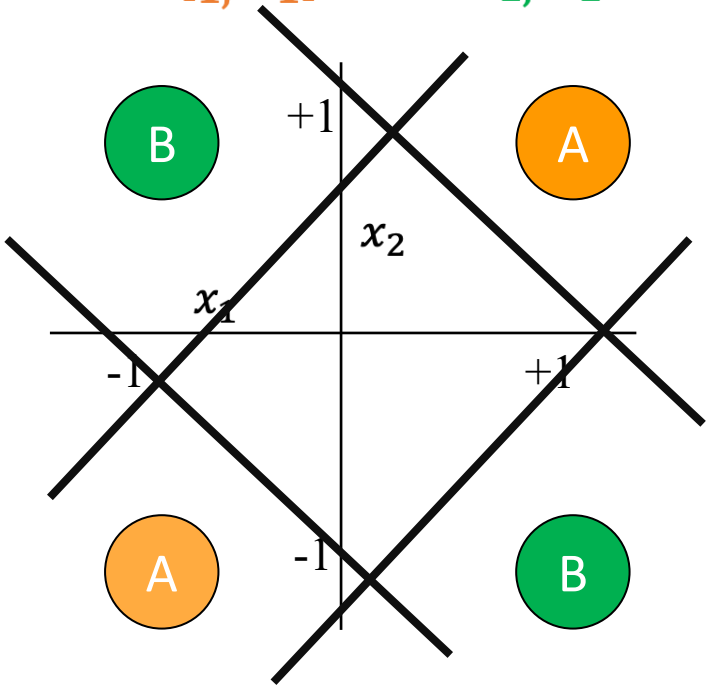
# Linearly separable data



# The XOR problem

$x^A = \begin{bmatrix} 1, -1 \\ 1, -1 \end{bmatrix}$ $x^B = \begin{bmatrix} -1\ 1 \\ 1, -1 \end{bmatrix}$



# Multi-layer perceptron

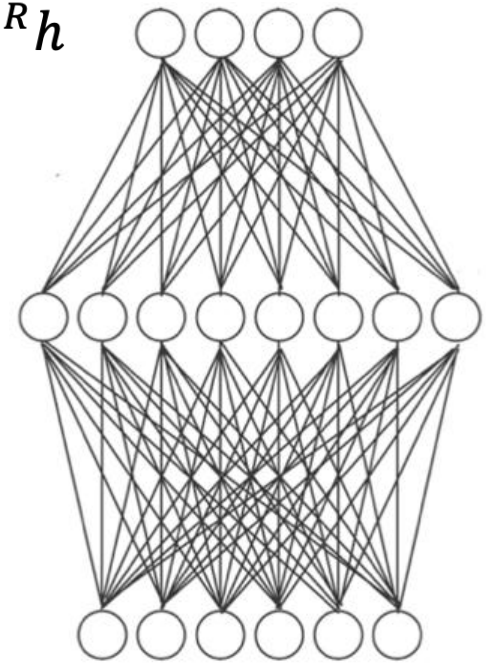
$y = W^R h$

$W^R$

$h = g(Wx)$

$W$

$x$



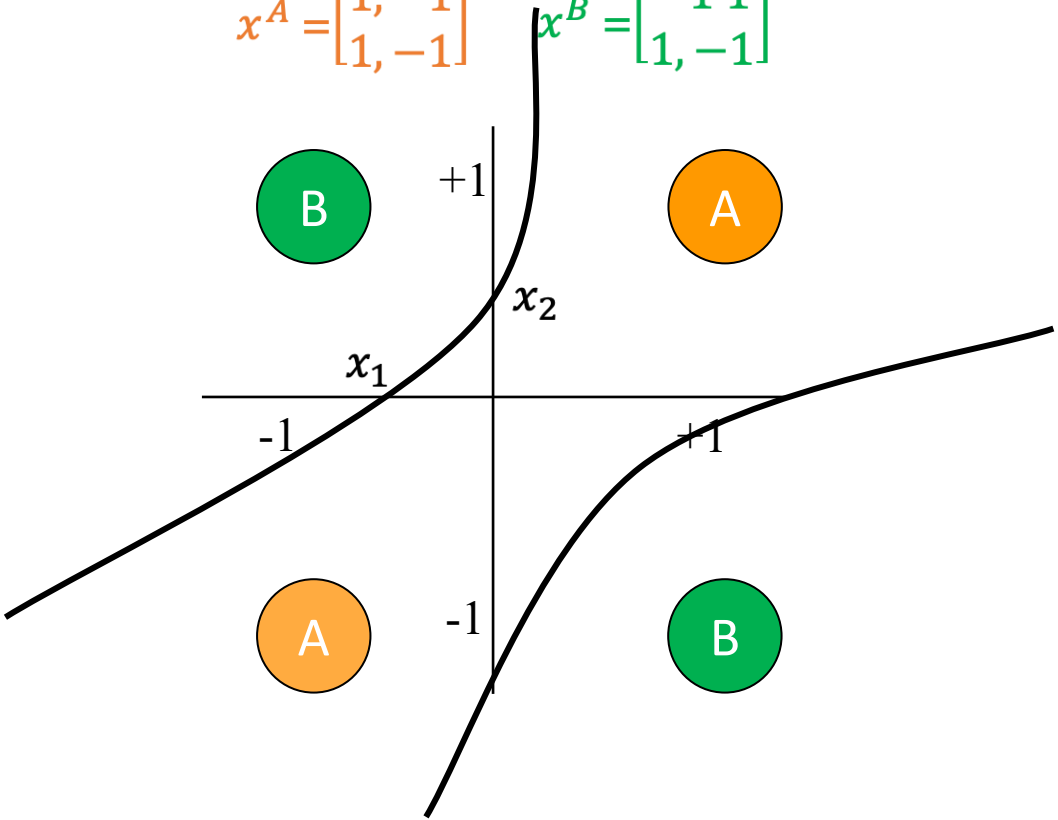
ReLU

$g(x) = \begin{cases} 0, x < 0 \\ x, x \geq 0 \end{cases}$

deep nonlinear network

# The XOR problem

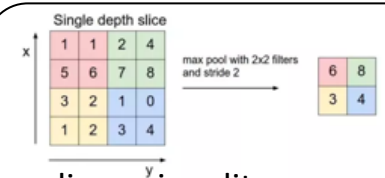
$x^A = \begin{bmatrix} 1, -1 \\ 1, -1 \end{bmatrix}$ $x^B = \begin{bmatrix} -1\ 1 \\ 1, -1 \end{bmatrix}$



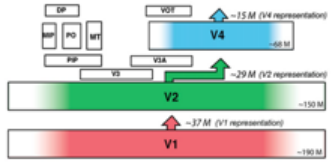
# Convolutional neural networks

$$b_i = \frac{a_i}{\left(k + \alpha \cdot \sum a_j^2\right)^\beta}$$

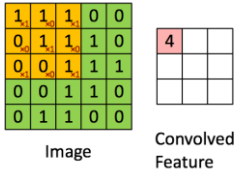
divisive normalisation



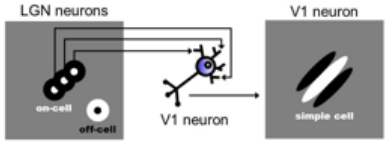

dimensionality reduction



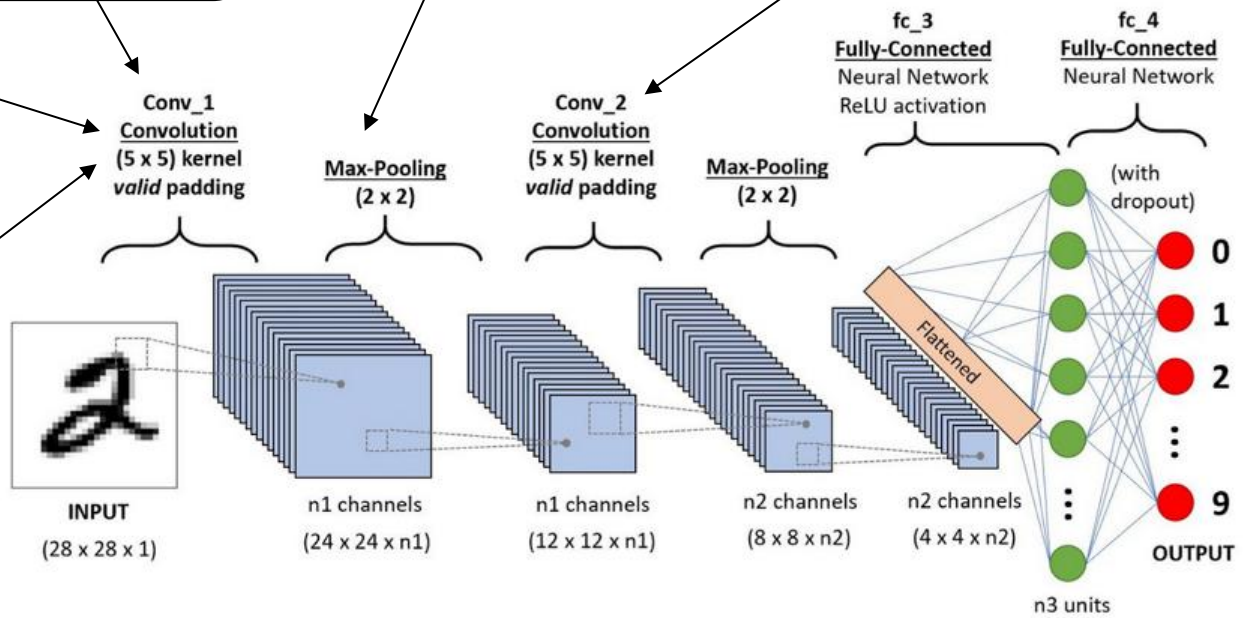

layerwise depth




convolution




local receptivity




```
import torch, torch.nn as nn

x = torch.tensor([[1.0], [2.0]])
y = torch.tensor([[3.0], [5.0]])

model = nn.Linear(1, 1)
```