Adaptive Intelligence

Prof. Eleni Vasilaki

“ADAPTIVE INTELLIGENCE is an innate ability seen in most human beings from birth.”

–Psychology Dictionary
https://psychologydictionary.org/adaptive-intelligence/

Computational Neuroscience

Machine Learning

Understanding Learning

“The fields of neuroscience and artificial intelligence (AI) have a long and intertwined history. […] we argue that better understanding biological brains could play a vital role in building intelligent machines.”

–Demis Hassabis et al, Deepmind, Neuron Review 2017

Why neuroscience?

- “It provides a rich source of inspiration for new types of algorithms and architectures.”

- “If a known algorithm is subsequently found to be implemented in the brain, then that is strong support for its plausibility as an integral component of an overall general intelligence system.”

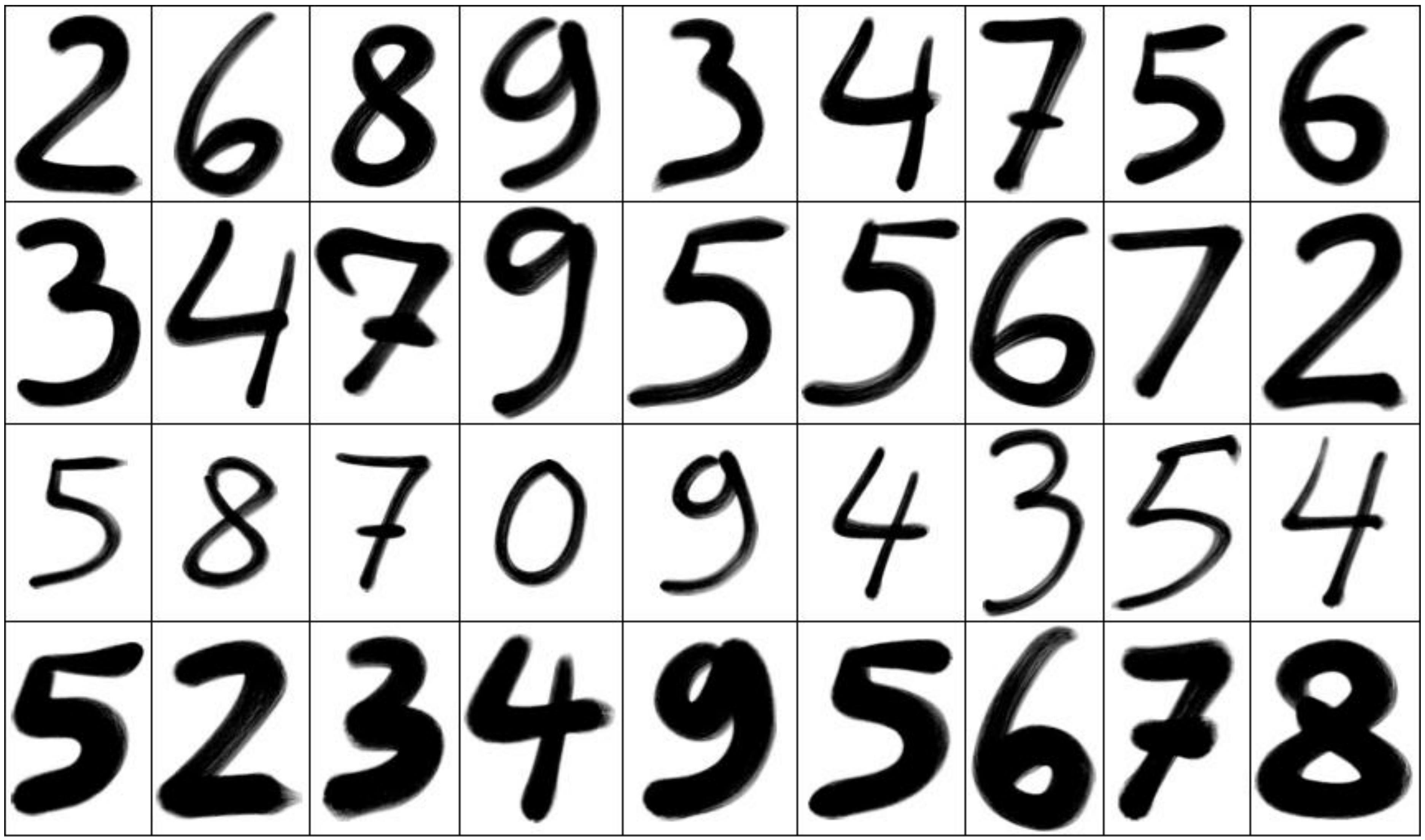
–Demis Hassabis et al, Deepmind, Neuron Review 2017

Machine Learning

Types of Learning

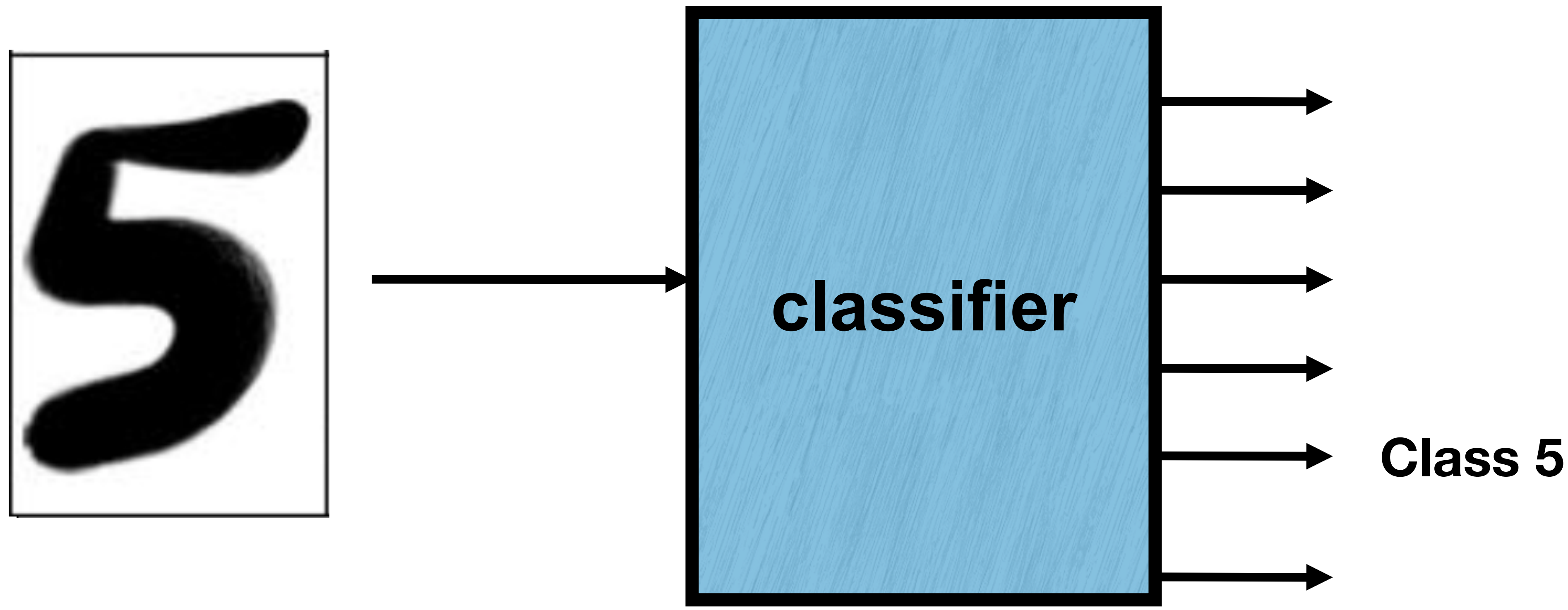
- Supervised
- Unsupervised
- Reinforcement

Supervised Learning

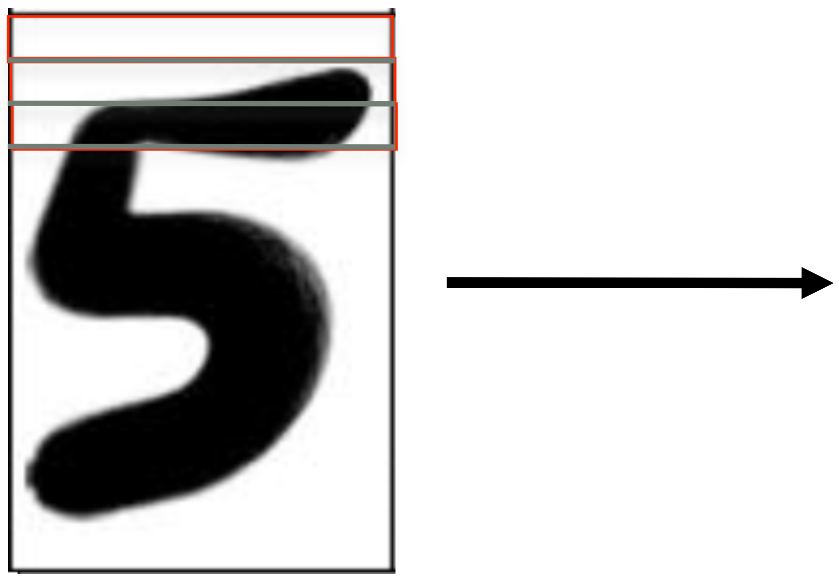


MNIST Database

Supervised Learning



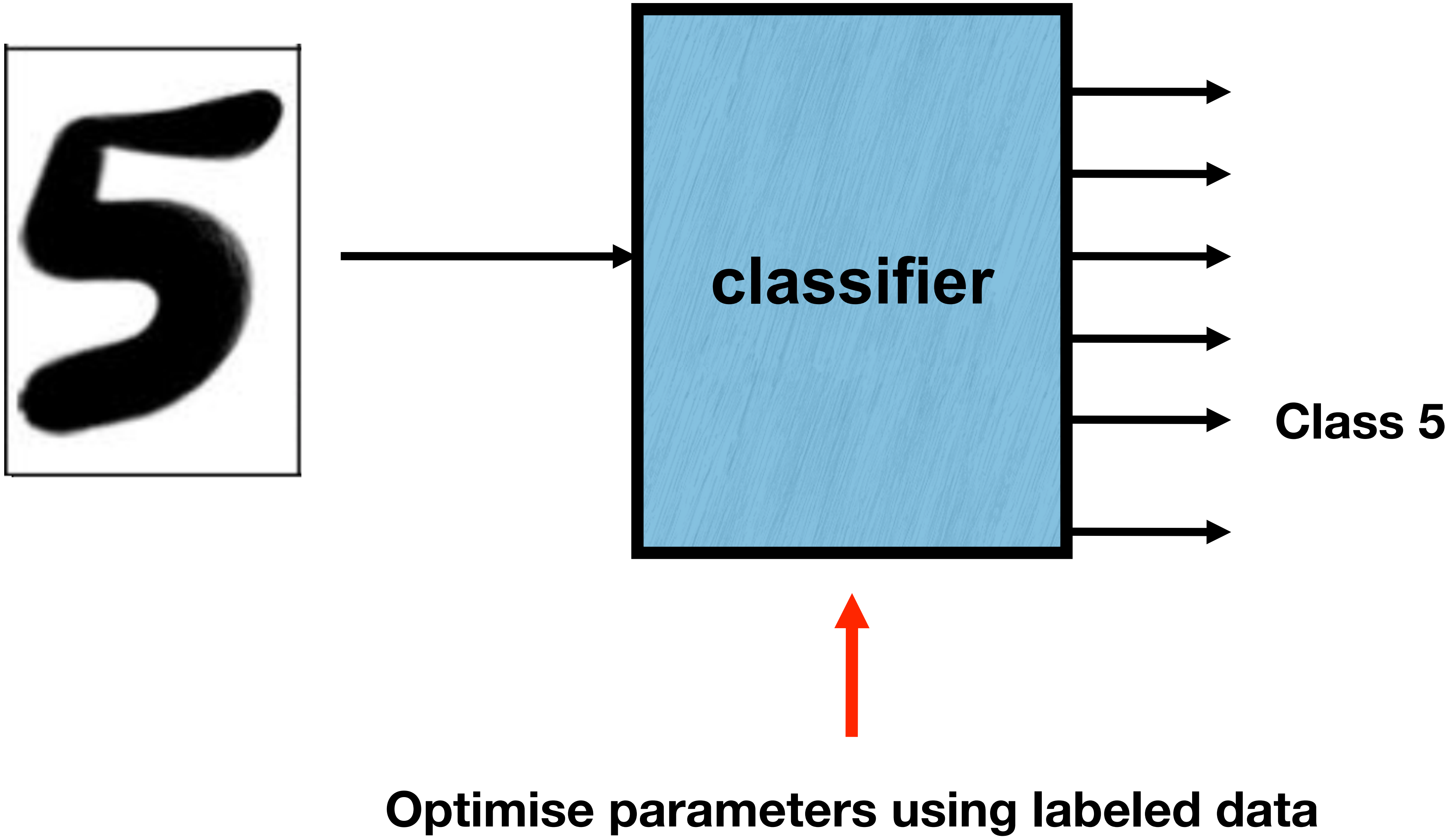
Supervised Learning



x^n

1D structure

Supervised Learning



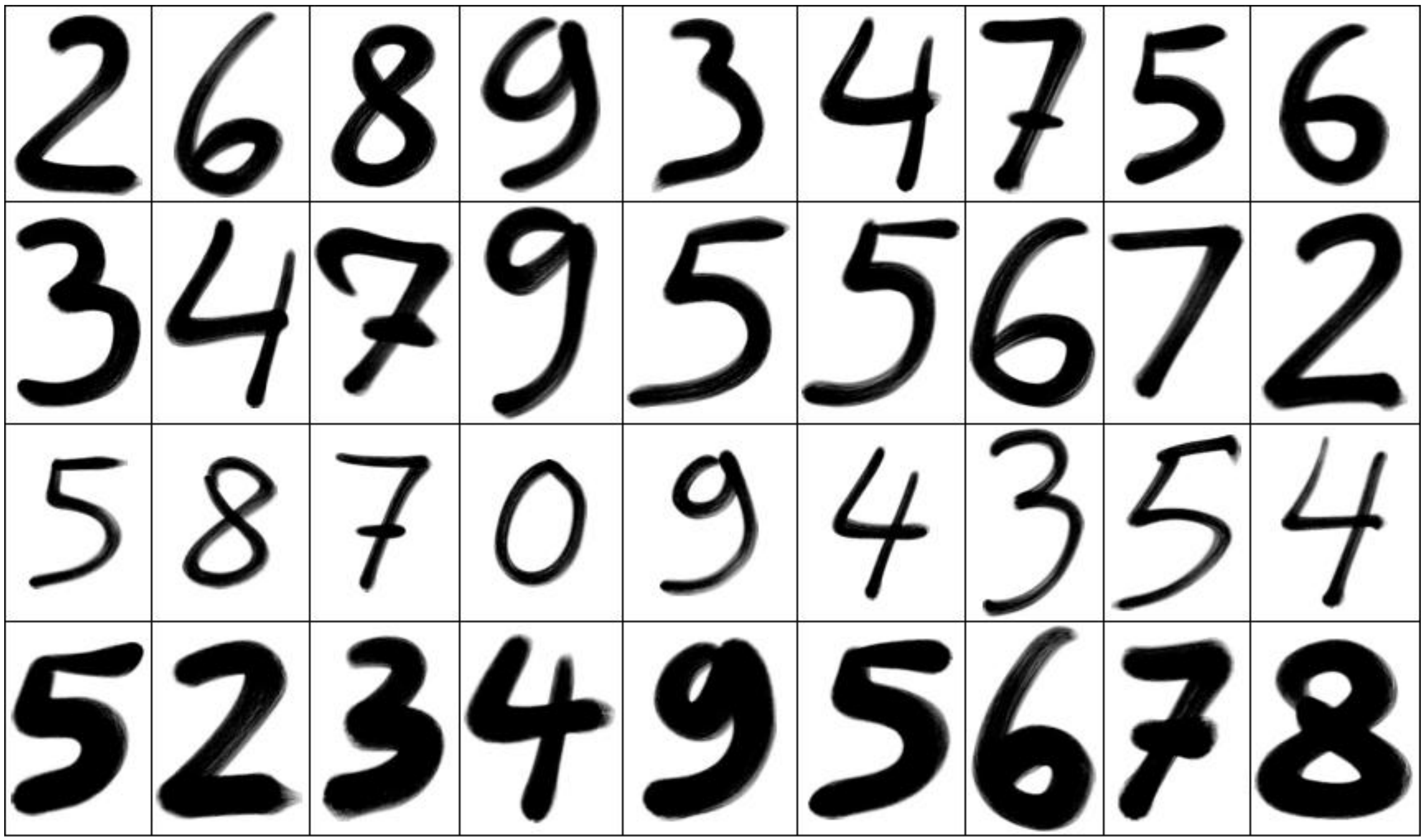
Supervised Learning

$$\{ (x^1, t^1)(x^2, t^2) \cdots (x^p, t^p) \}$$

- An expert (supervisor) has provided labels for the data (e.g. class 1, class 2 etc).
- Parameters are updated so that the total error, evaluated using the training (labeled) data, is minimised.
- The system, if correctly trained, is expected to perform also with unseen samples from the same classes.

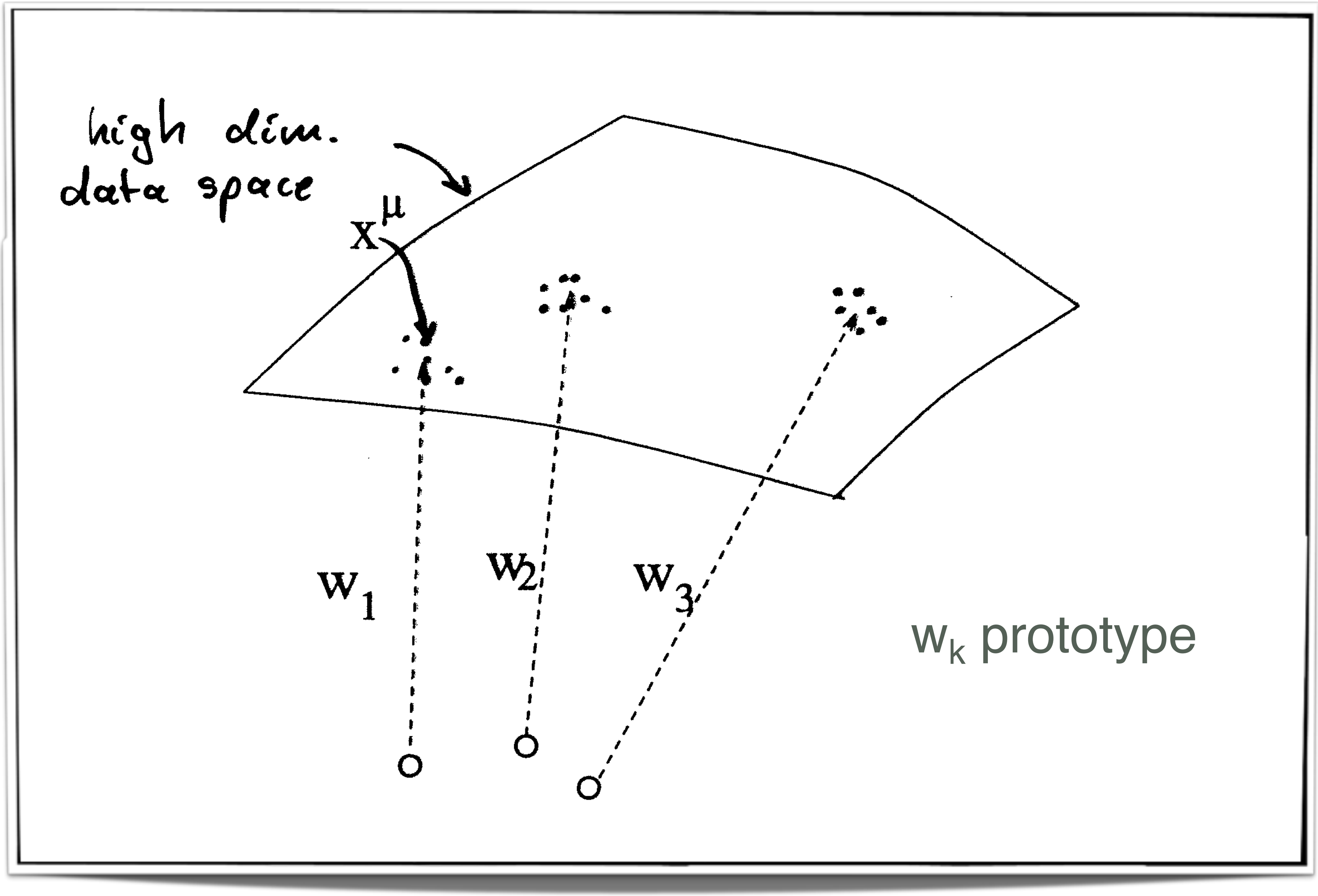
Unsupervised Learning

Unsupervised Learning



No labels!

Unsupervised Learning



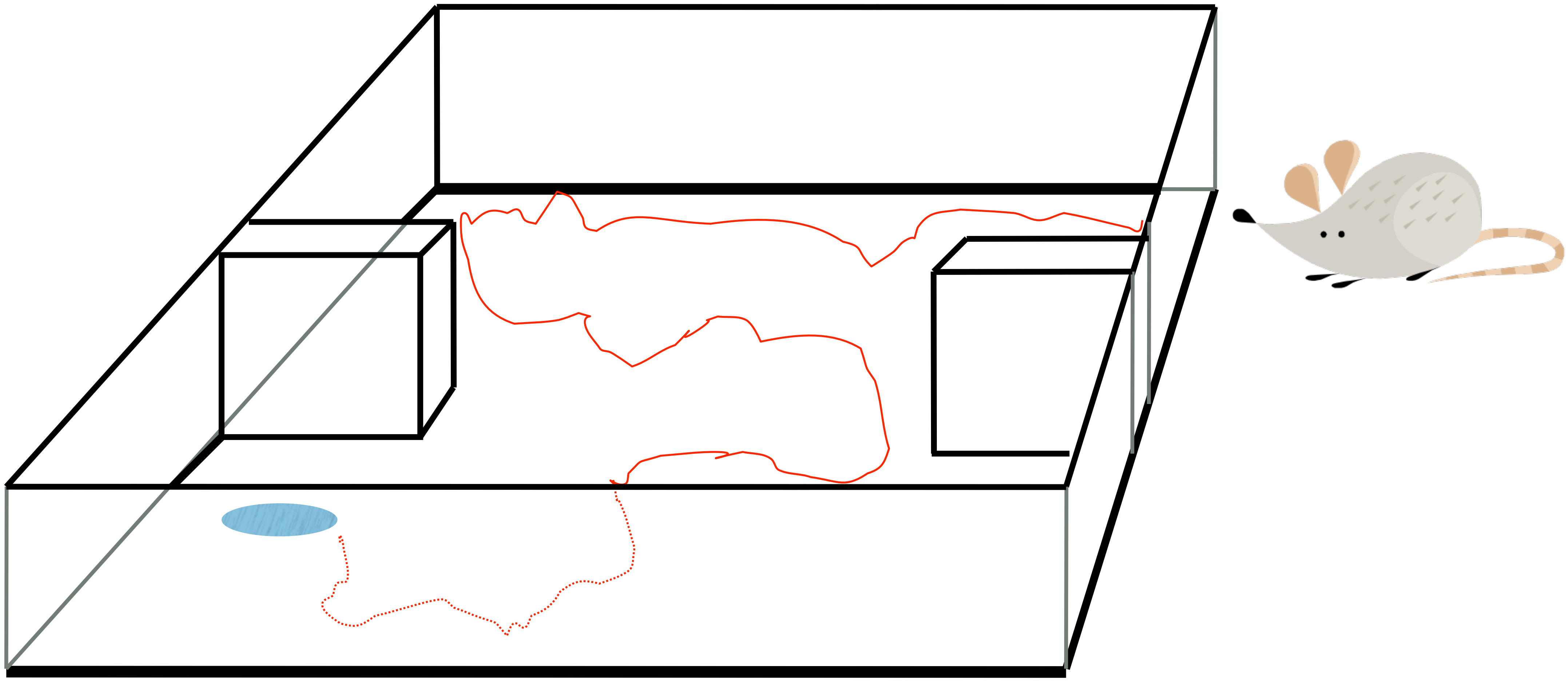
Unsupervised Learning

$$\{\ x^1, x^2, x^3, \quad \cdots \quad x^p\ \}$$

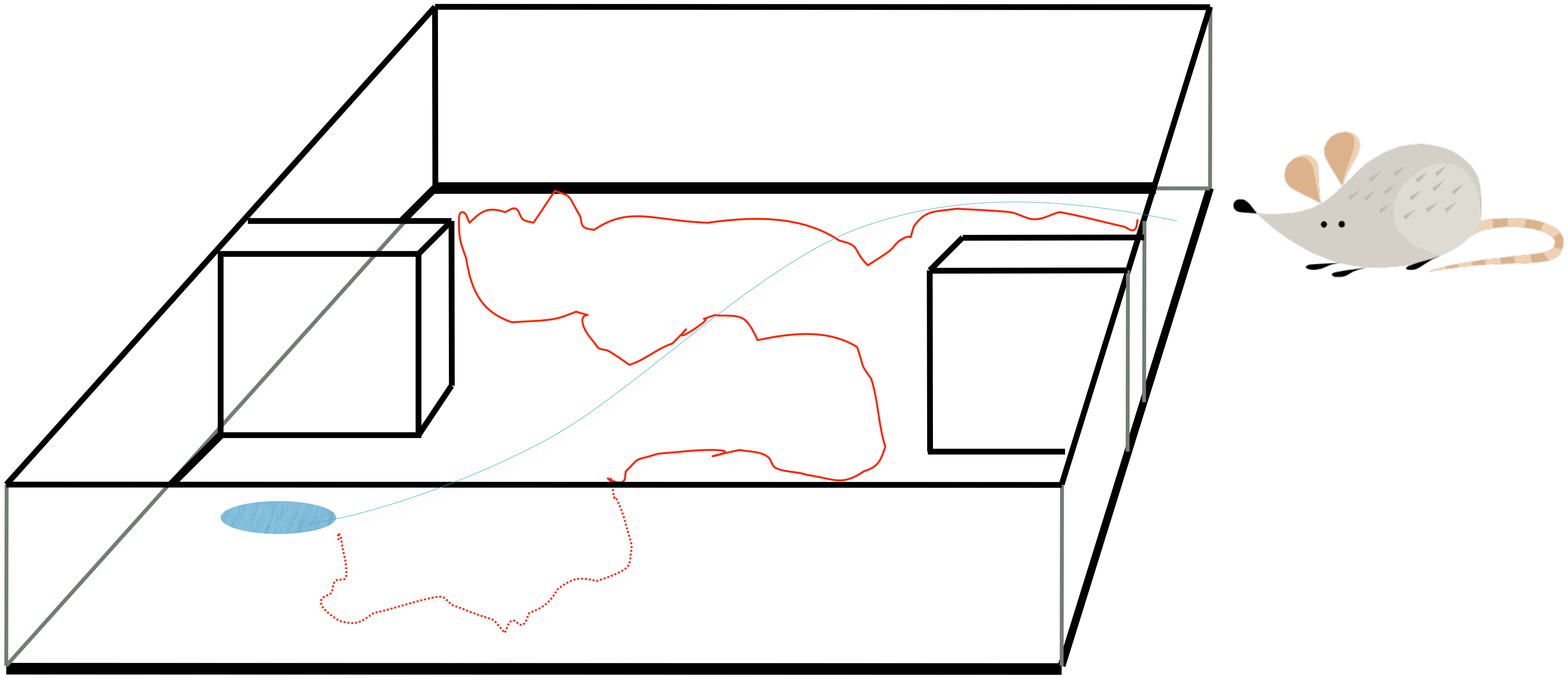
- Just data - NO labels!
- Detect intrinsic structure in the data.
- Leads to clusters (not classes).

Reinforcement Learning

Reinforcement Learning



Reinforcement Learning



Reinforcement Learning

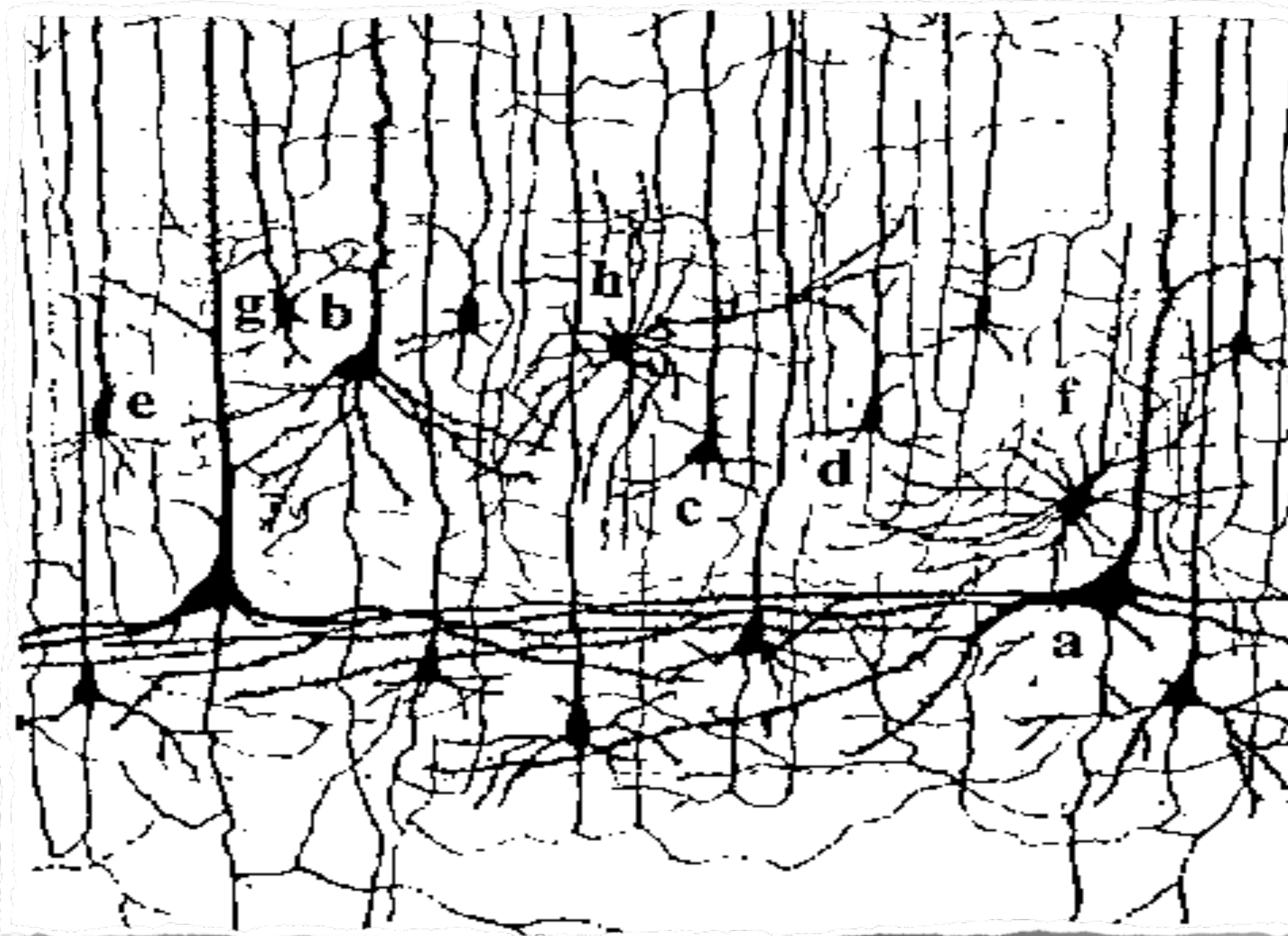
$$\{ \ (x^1, a^1, r^{11}) \ (x^1, a^2, r^{12}) (x^2, a^1, r^{21}) \ \cdots \ \}$$

- An agent receives an input/stimulus.
- Choice of actions.
- Reward upon correct action (or actions - it may not be immediate!).
- Parameters are updated so that the total Reward is maximised.

Task	Supervised	Reinforcement	Unsupervised
Riding a bicycle			
Driving a car			
Deciphering the Rosetta Stone			
Navigating into an unknown building			
Training a tiger to jump through a ring of fire			
Classifying musical pieces to different music genres			
After breaking the dominant hand, learn to use the other one for all tasks			
Proposing books to buyers according their previous choices (e.g. on-line bookshops)			
Developing social skills			
Building a classifier for handwritten digits			

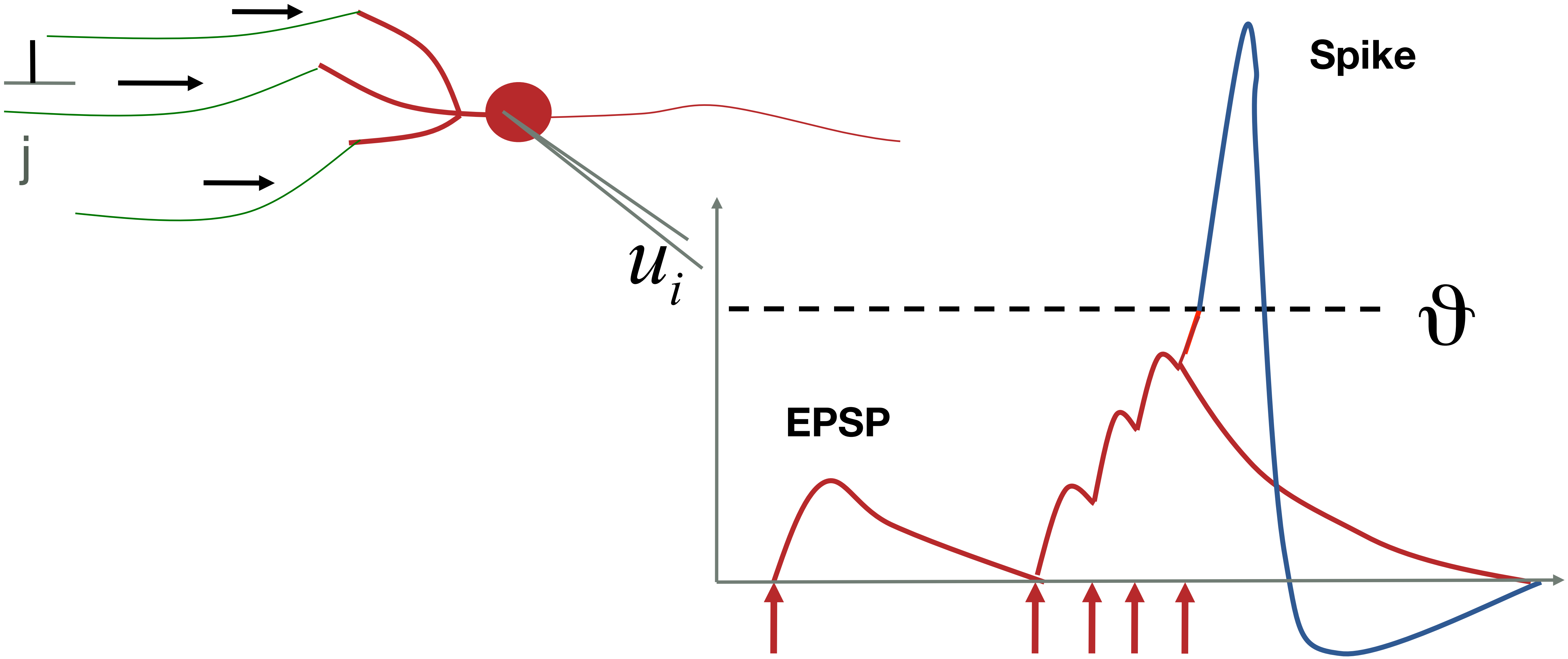
Course Overview

- Unsupervised Learning in Neural Networks
    - Hebb rule, Oja’s rule, PCA, Competitive Learning.
- Reinforcement Learning
    - SARSA Learning



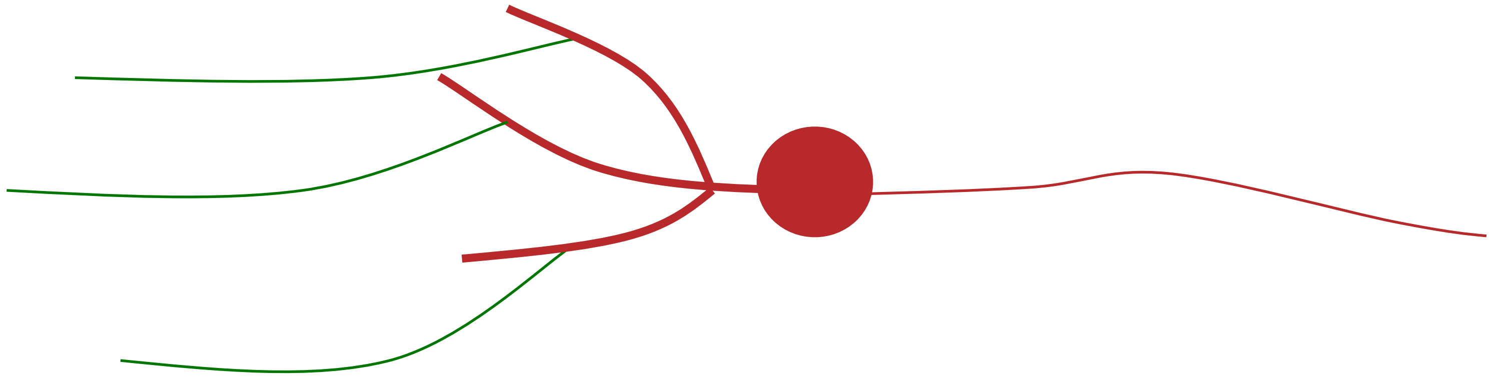
1mm^3 : 10,000 Neurons 3km wires

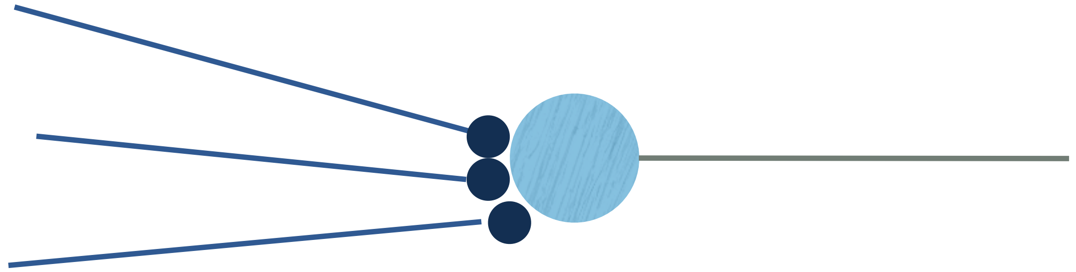
Real Neurons



Spike reception: EPSP

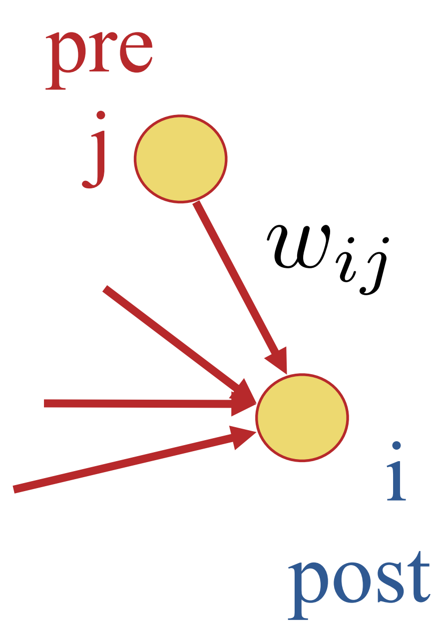
Artificial Neurons





$$\nu_i = g\left(\sum_j w_{ij}\nu_j\right)$$

The Hebbian rule



Correlations

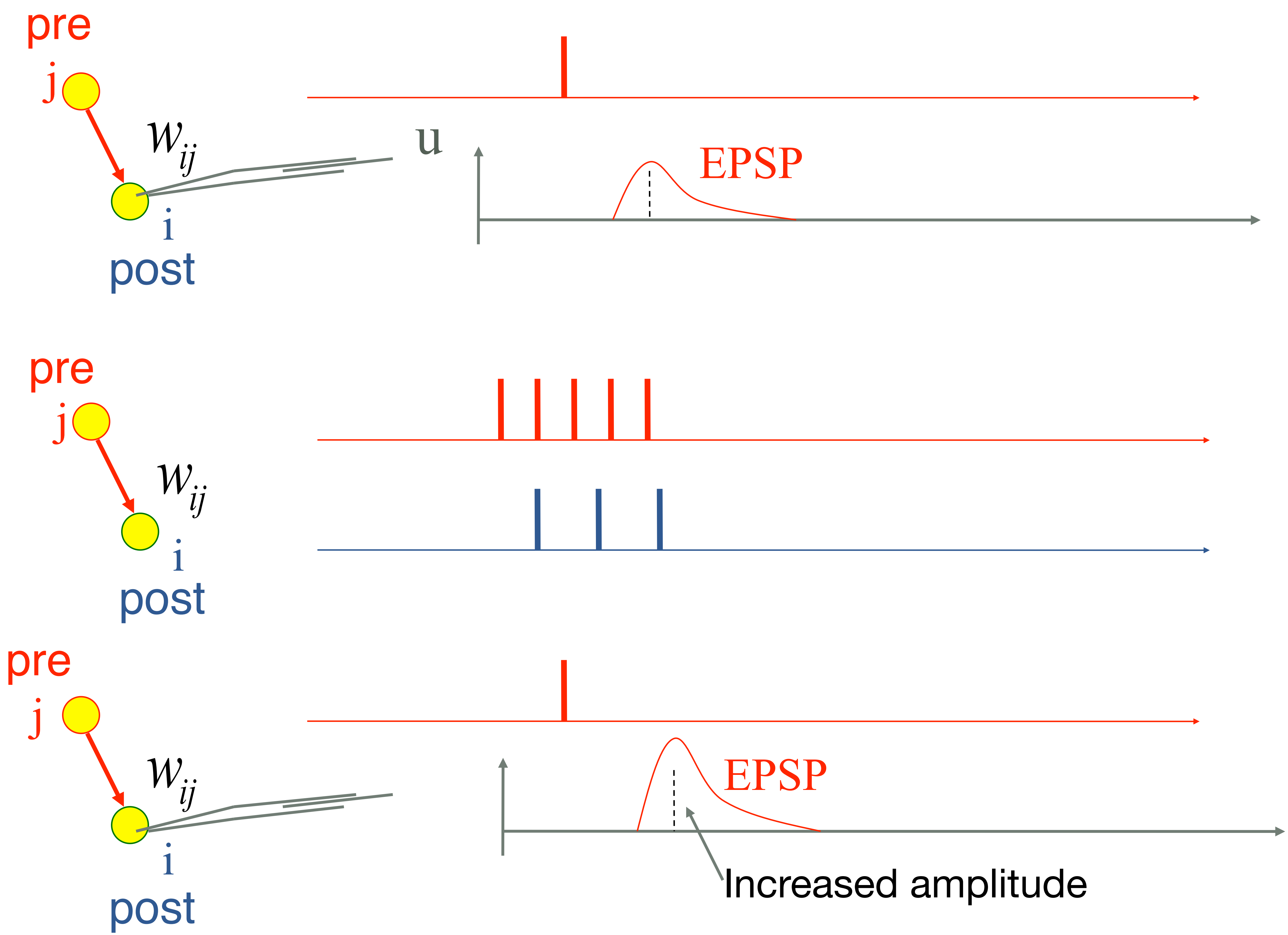
Donald Hebb

“When an axon of cell A is near enough to excite cell B or repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.”

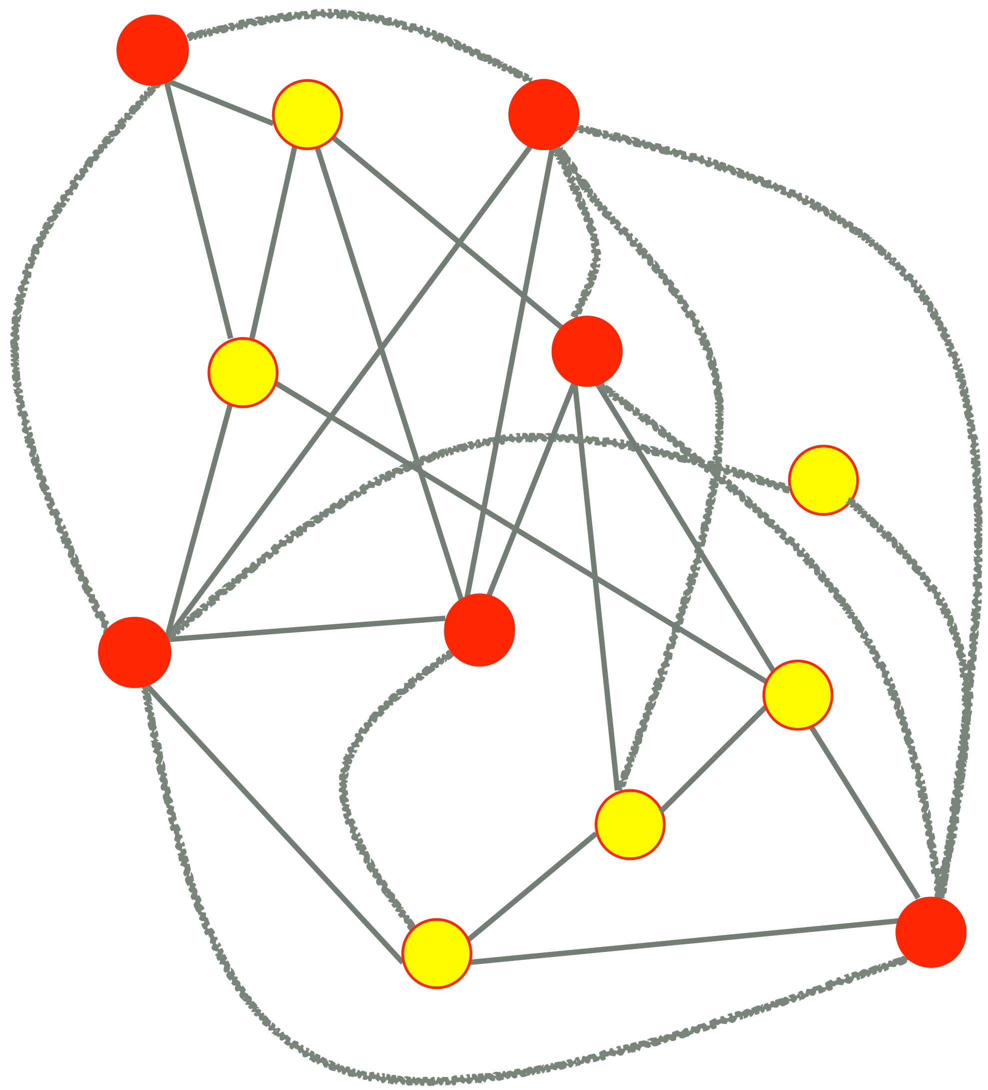
–Donald Hebb, 1949

“Neurons that fire together wire together.”

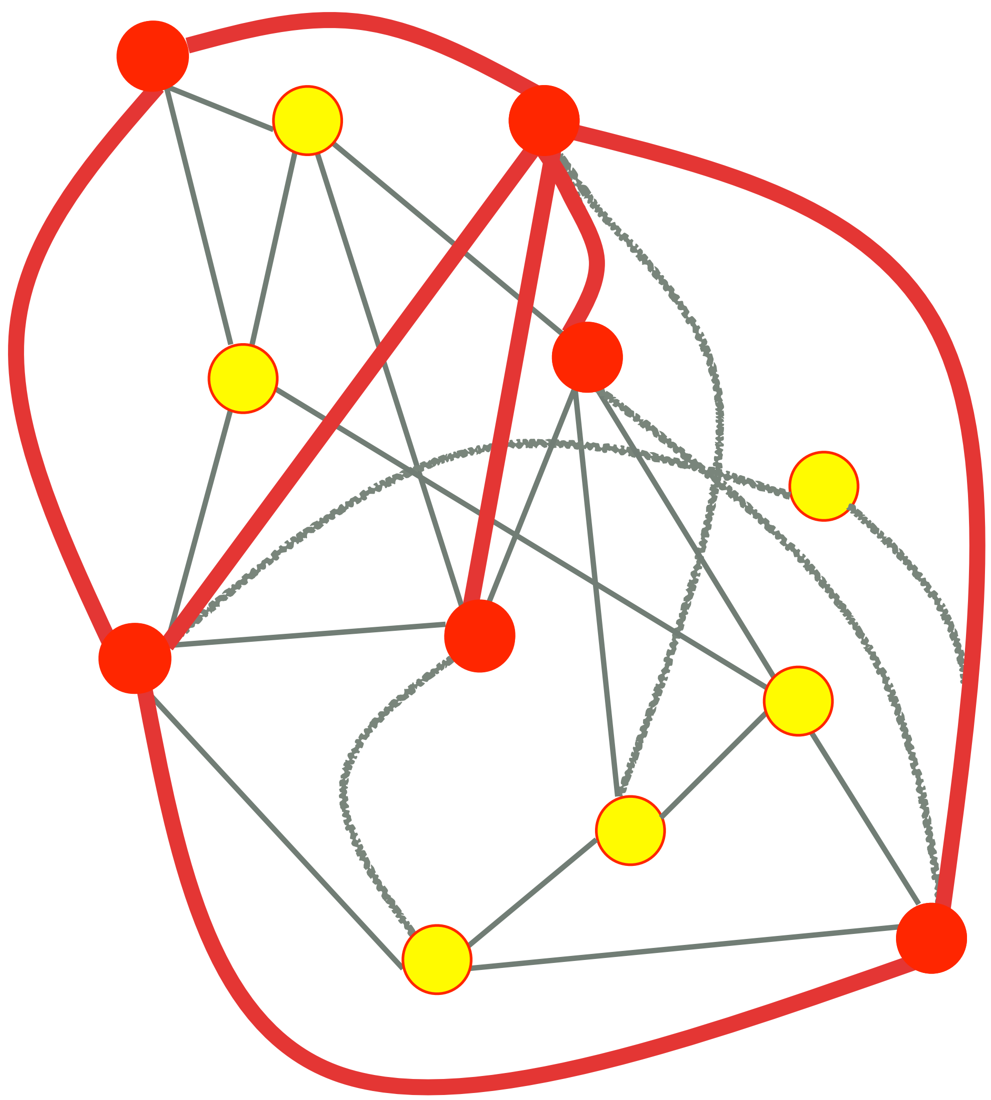
–A compact form of Donald Hebb’s postulate


pre
j
w_{ij}
u
EPSP
i
post
pre
j
w_{ij}
i
post
pre
j
w_{ij}
EPSP
i
post
Increased amplitude

Hebbian Learning



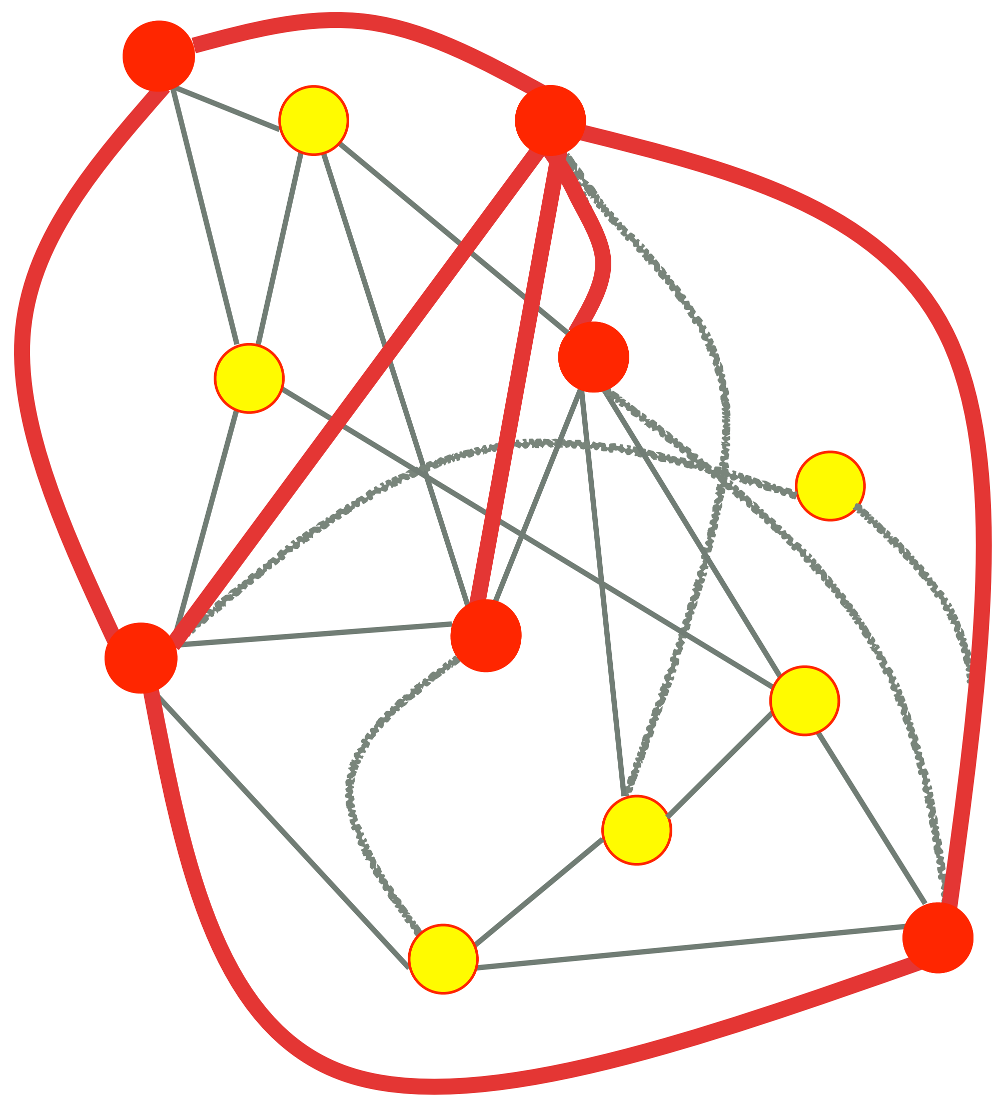
Hebbian Learning



item memorized

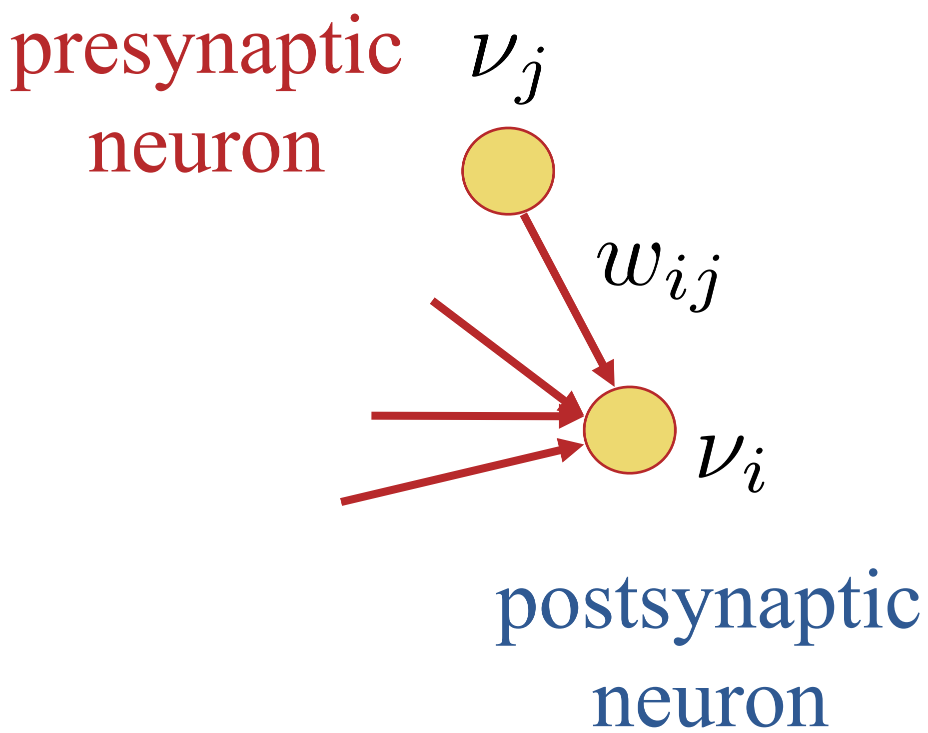
Hebbian Learning

Recall:
Partial info



item recalled

The “minimal” Hebbian Rule



In equations

$$\Delta w_{ij} = w_{ij}^{new} - w_{ij}^{old} = \alpha \nu_i \nu_j$$

w_{ij} : weight from neuron j to neuron i

Δw_{ij} : weight change

α : learning rate, >0

Learning rule notation

Discrete time

$$\Delta w_{ij} = f\left(\nu_i, \nu_j, w_{ij}\right)$$

$$w_{ij} \to w_{ij} + f(\nu_i, \nu_j, w_{ij})$$

Continuous time

$$\frac{dw_{ij}}{dt} = g(\nu_i, \nu_j, w_{ij})$$

Hebbian rules

$$\Delta w_{ij} = \alpha \nu_i \nu_j$$

Common "ingredient"

$$\Delta w_{ij} = \alpha \nu_i \nu_j$$

$$\Delta w_{ij} = \alpha \nu_i \nu_j - c$$

$$\Delta w_{ij} = \alpha (\nu_i - \theta) \nu_j$$

$$\Delta w_{ij} = \alpha (\nu_i - \theta)(\nu_j - \theta)$$

Which rules are Hebbian?

Variable definitions. x: stimulus (neuronal input), y: neuronal activity, w: synaptic weight, ϵ: small positive number (learning rate), r : reward. Unless otherwise stated, x,y are mean firing rates.

1. Rescorla-Wanger (reward predicting) rule. $w \to w + \epsilon \delta x$, where $\delta = r - y$, $x \in \{0, 1\}$.
2. Homeostatic rule. $w \to w + \epsilon\,(y - \theta)$, where θ is the homeostatic threshold.
3. Node perturbation. $w \to w + \epsilon r y \xi$, where ξ represents noise (random variable drawn from a Gaussian distribution).

Which rules are Hebbian?

5. Sejnowski rule. $w \rightarrow w + \epsilon\,(y - \langle y \rangle)\,(x - \langle x \rangle)$, with $\langle . \rangle$ being the mean activity of the signals y and x across time.

6. Oja rule. $w \rightarrow w + \epsilon y\,(x - yw)$.

7. Learning in Hopfield networks. $w \rightarrow w + \epsilon yx$, with $x, y \in \{-1, 1\}$.

8. Delta rule. $w \rightarrow w + \epsilon \delta x$, where $\delta = t - y$, with t being the target for input x.

9. Associative Reward Inaction $w \rightarrow w + \epsilon r\,(y - P(y))\,x$, where $P(y)$ the probability of y to be active, $y \in \{0, 1\}$.

Practicalities

- Warning: Math intensive module!
- Evaluation: Lab 35%, Exam 65%
- Python or Matlab, you are expected to know one of the two already!

Acknowledgements

- This module is an adaptation of the MSc module "Unsupervised and Reinforcement Learning in Neural Networks" by Prof. Wulfram Gerstner at EPFL, Switzerland.

Bibliography

- Neuronal Dynamics, by Gerstner et al.
- Introduction to the Theory of Neural Computation" by Hertz, Krogh & Palmer
- Reinforcement Learning: An Introduction" by Sutton & Barto
- Scholarpedia

Thank you!