Neural Networks

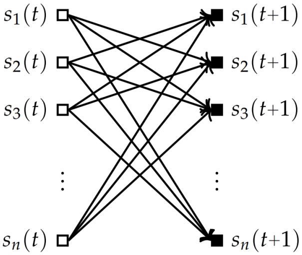
9. Hopfield Networks

Center for Cognitive Science
Department of Applied Informatics
Faculty of Mathematics, Physics and Informatics
Comenius University in Bratislava

Thursday 18$^{\text{th}}$ April, 2024

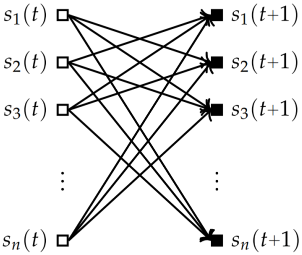
Hopfield network

- **network:** n neurons with ± 1 - threshold activation
- **state:** one value for each neuron:
  $\mathbf{s} \in \{\pm 1\}^n$
- **weights:** connections for each pair of neurons:
  $\mathbf{W} \in \mathbb{R}^{n \times n}$
  - no reflexive weights - diagonal is empty (zero)



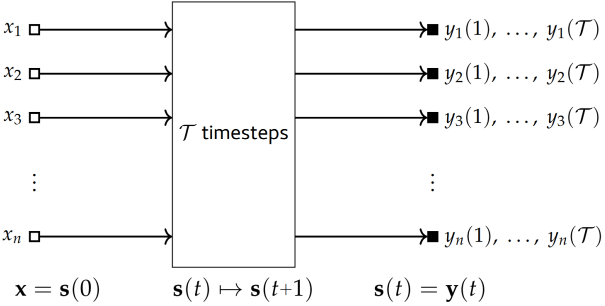
Insider view: time slice

- what happens inside the network - time step $t \rightarrow t+1$



- computation of new state:
$$s_i(t+1) = f(\mathbf{w}_i \cdot \mathbf{s}(t))$$

Outsider view



- neuron activations are initialized to the outputs
- network updates its states $\mathbf{s}(t)$ for τ timesteps
- outputs at the time t are the neuron activations at t

Operation modes

- only the current state influences the next state (Markov property)

- **synchronous (parallel) dynamics:**
  - all neurons change state at once
    $\mathbf{s} \rightarrow \mathbf{s}'$

- **asynchronous (sequential) dynamics:**
  - only one neuron "recomputes" at a time
    $\mathbf{s}_i \rightarrow \mathbf{s}'_i$

note: we use $(\mathbf{s}, \mathbf{s}')$ and $(\mathbf{s}(t), \mathbf{s}(t+1))$ interchangeably

Transition Functions

- conventional net_i for the i-th neuron:
  $$net_i = \mathbf{w}_i \cdot \mathbf{s}$$
- **deterministic transition:** "$\pm$ sign" function
  $$s_i' = sgn^*(net_i)$$
  $$sgn^*(net_i) = \begin{cases} +1 & net_i \geq 0 \\ -1 & net_i < 0 \end{cases}$$
- **stochastic (probabilistic) transition:**
  $$P[s_i' = 1] = \frac{1}{1+e^{-\beta \cdot net_i}}$$
  - depends on β -inverse temperature: $\beta = 1/T$

State space dynamics

- energy of a state:
  $$E_{\mathbf{W}}(\mathbf{s}) = -\frac{1}{2}\sum_j \left(\sum_{i \neq j} w_{i,j} s_i s_j\right)$$
- transitions $t \to t+1$: moving in state space
- computation $\approx$ relaxation to states with lower energy
- possible outcomes (for deterministic synchronous):
  - fixed point (true or false attractor):
    $$\mathbf{s}(t) = \mathbf{s}(t-1)$$
  - cycles (even length):
    $$\mathbf{s}(t) = \mathbf{s}(t-2k); \quad \exists k \in \mathbb{N}$$

Hopfield Auto-associative Memory

- **patterns:** P points in n -dimensional space:
  $$\mathbf{x}^1, \mathbf{x}^2, ..., \mathbf{x}^P \quad \in \mathbb{R}^n$$
- we "store" patterns in the weight matrix
- we want the stored patterns to be energy minima
- **analytic training** ($\approx$ correlations):
  $$w_{i,j} = \begin{cases} \frac{1}{P} \sum_P x_i^P x_j^P & i \neq j \\ 0 & i = j \end{cases}$$

Task

- hopfield.py
  - training - compute $\mathbf{W}$ analytically
  - energy $E_{\mathbf{W}}(\mathbf{s})$
  - implement **asynchronous** dynamics
  - implement **stochastic** and **deterministic** transitions