Three Analog Neurons Are Turing Universal

Jiří Šíma

Institute of Computer Science
Czech Academy of Sciences

(Artificial) Neural Networks (NNs)

1. mathematical models of biological neural networks

- simulating and understanding the brain (The Human Brain Project)
- modeling cognitive functions

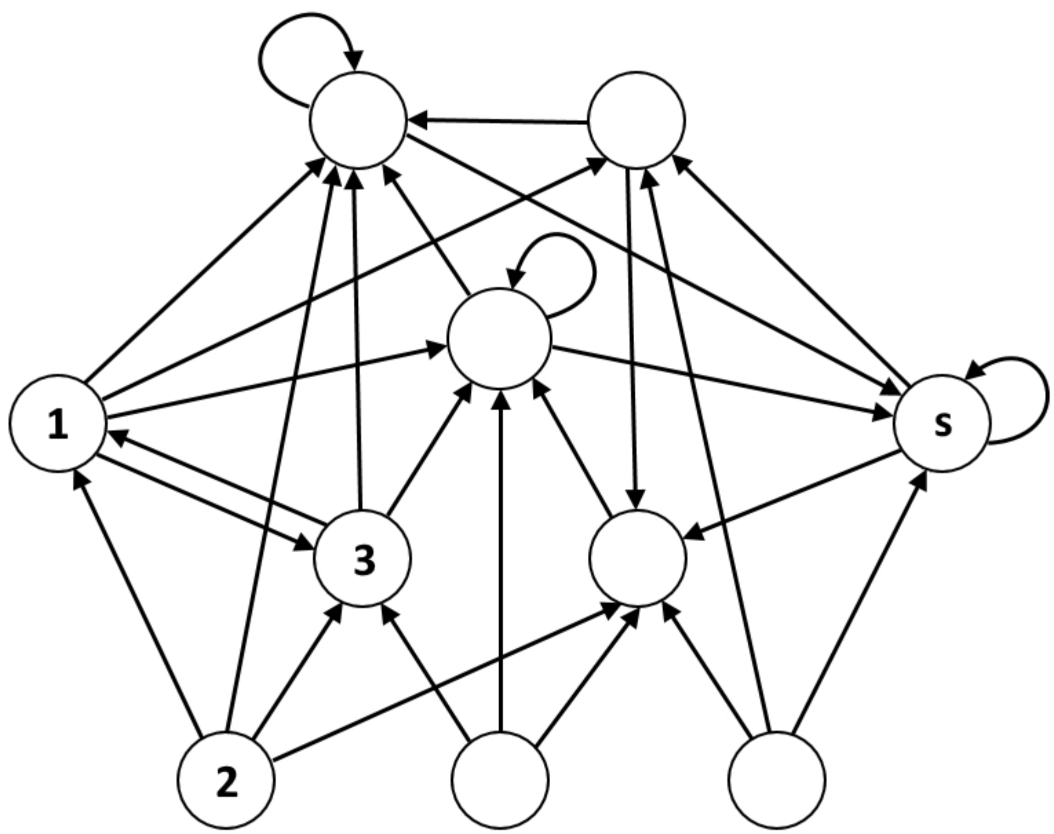
2. computing devices alternative to conventional computers

already first computer designers sought their inspiration in the human brain
(e.g., neurocomputer due to `Minsky, 1951`)

- common tools in machine learning or data mining (learning from training data)
- professional software implementations (e.g. `Matlab`, `Statistica` modules)
- successful commercial applications in AI (e.g. deep learning):
  computer vision, pattern recognition, control, prediction, classification, robotics, decision-making, signal processing, fault detection, diagnostics, etc.

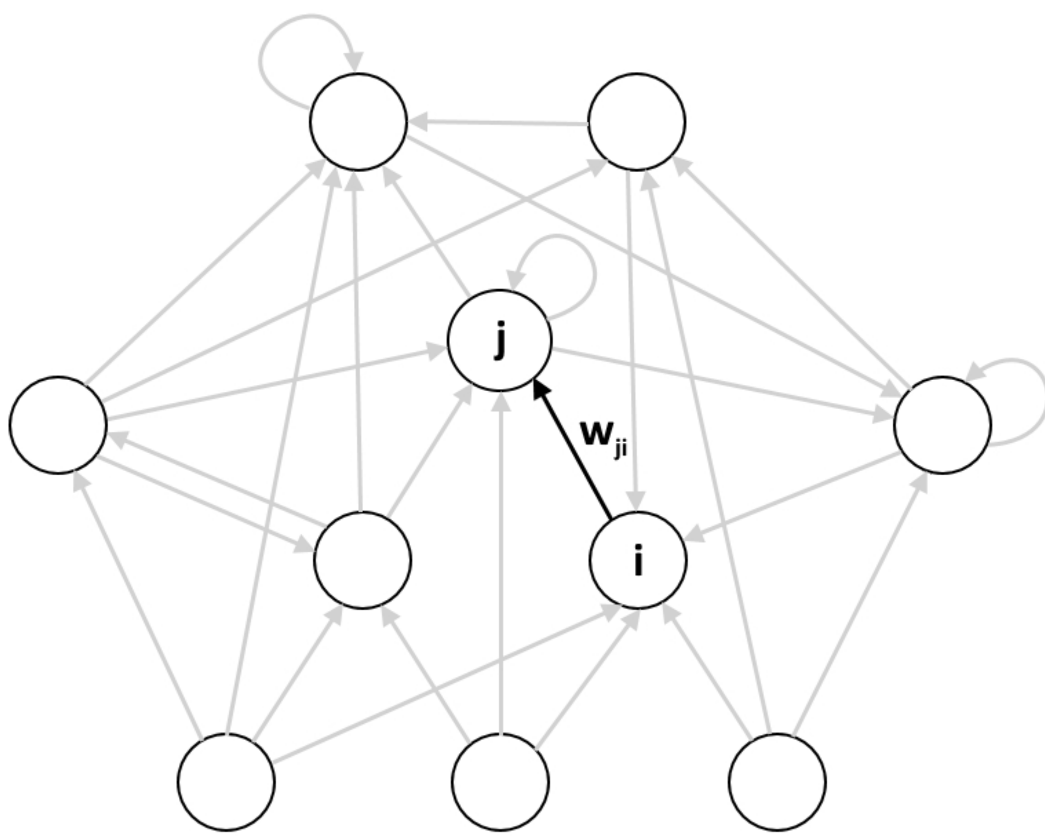
The Neural Network Model – Architecture

s computational units (neurons), indexed as $V = \{1, \ldots, s\}$, connected into a directed graph (V, A) where $A \subseteq V \times V$



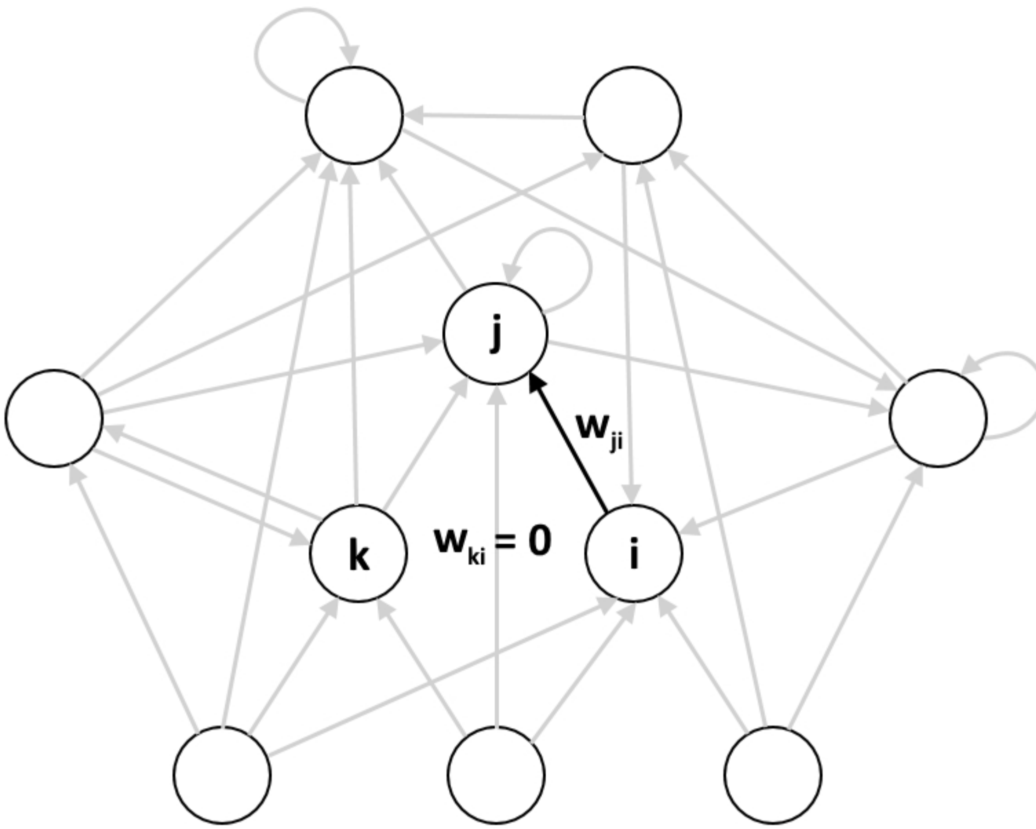
The Neural Network Model – Weights

each edge $(i, j) \in A$ from unit i to j is labeled with a real weight $w_{ji} \in \mathbb{R}$



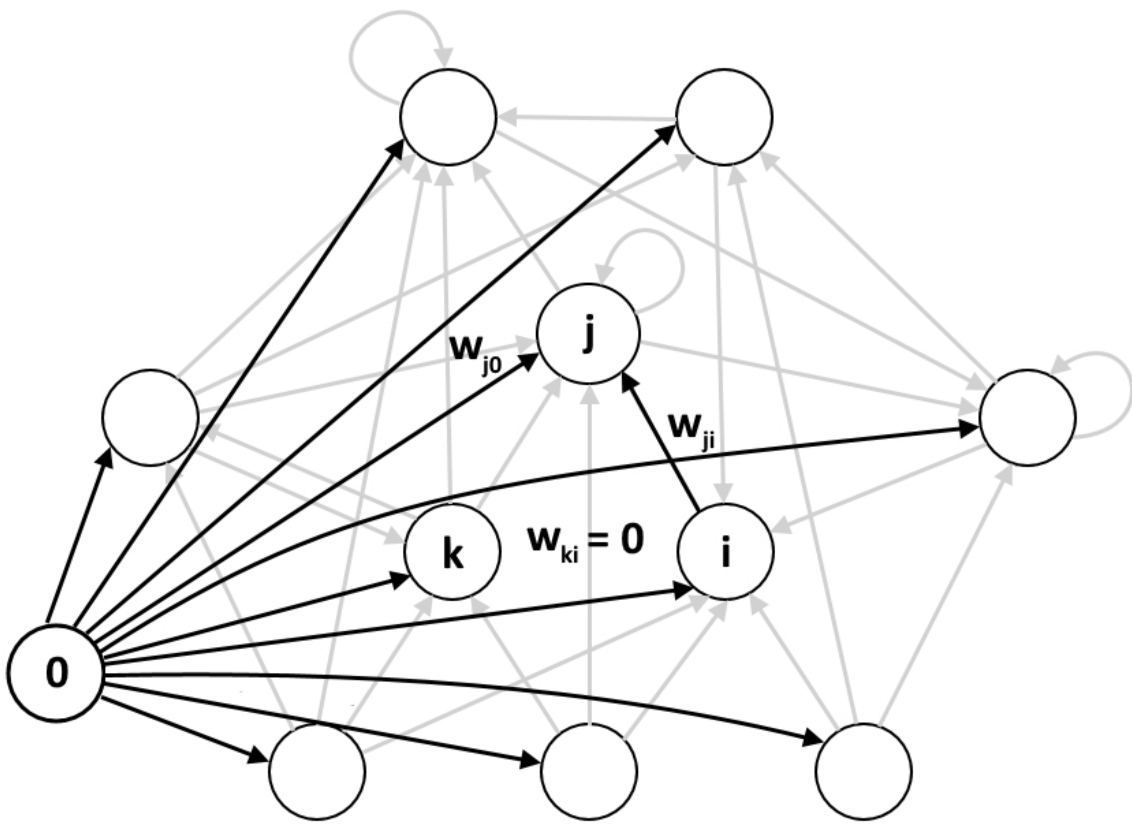
The Neural Network Model – Zero Weights

each edge $(i, j) \in A$ from unit i to j is labeled with a real weight $w_{ji} \in \mathbb{R}$
$(w_{ki} = 0$ iff $(i, k) \notin A)$



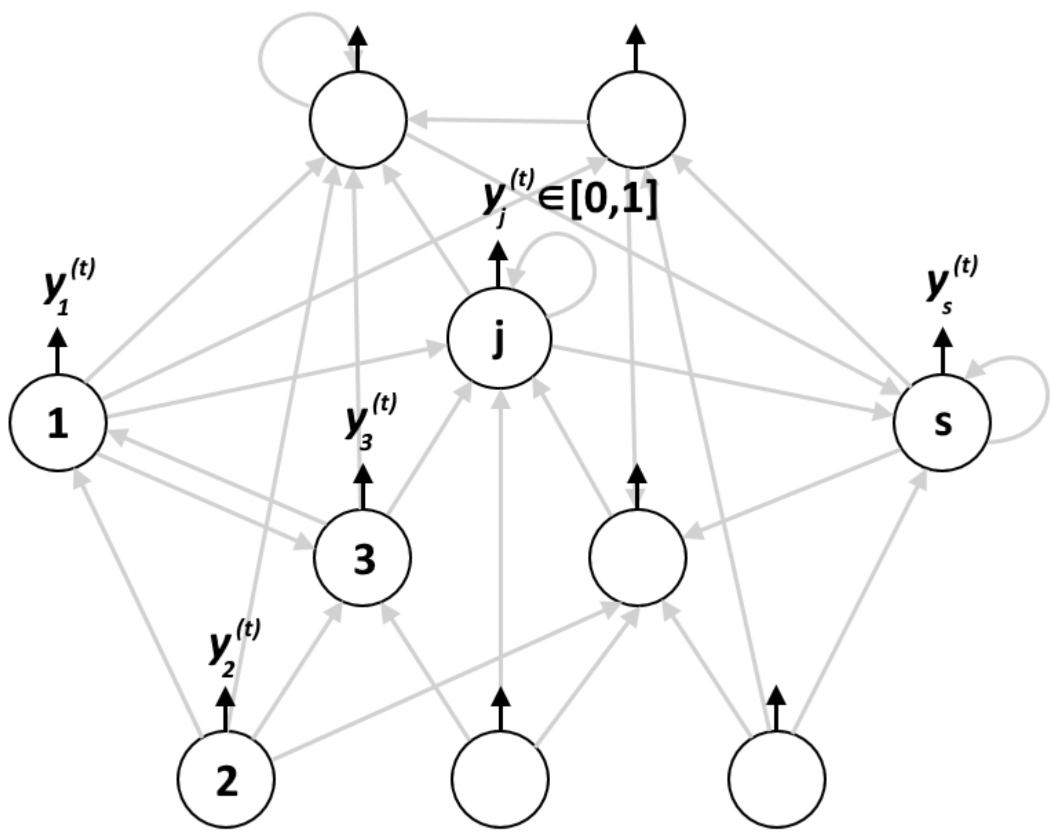
The Neural Network Model – Biases

each neuron $j \in V$ is associated with a real bias $w_{j0} \in \mathbb{R}$
(i.e. a weight of $(0, j) \in A$ from an additional formal neuron $0 \in V$)



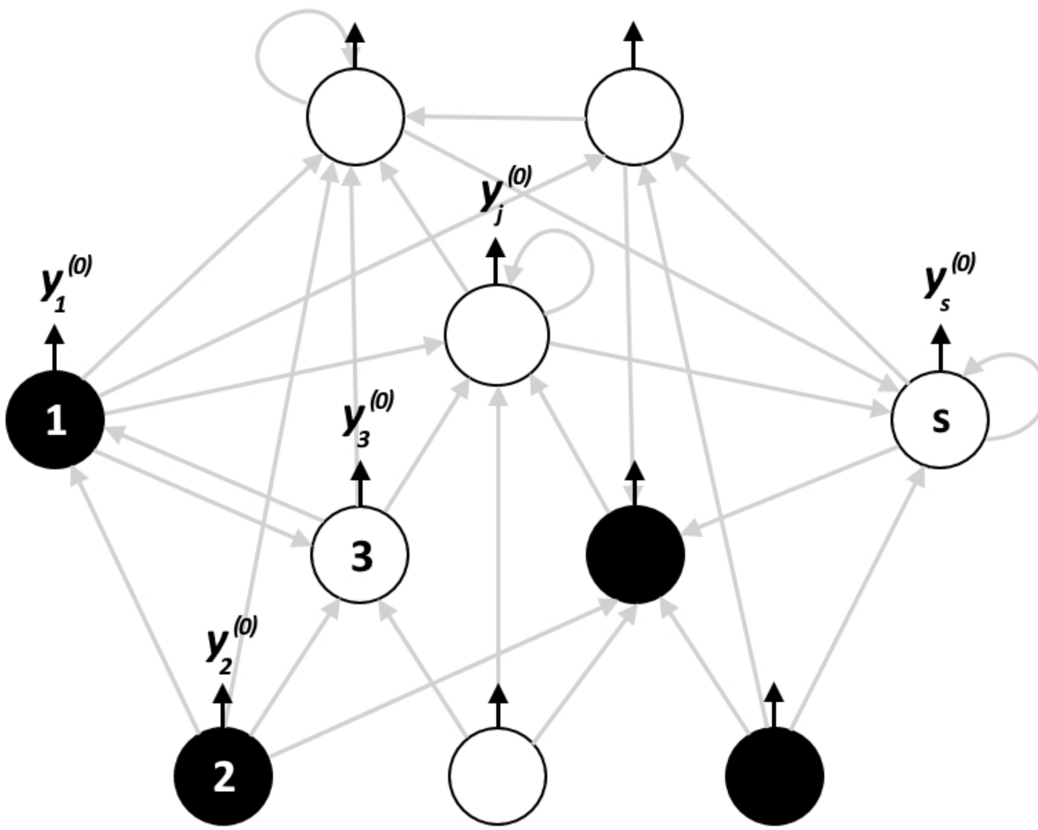
Discrete-Time Computational Dynamics – Network State

the evolution of global network state (output) $\mathbf{y}^{(t)} = (y_1^{(t)}, \ldots, y_s^{(t)}) \in [0,1]^s$
at discrete time instant $t = 0, 1, 2, \ldots$



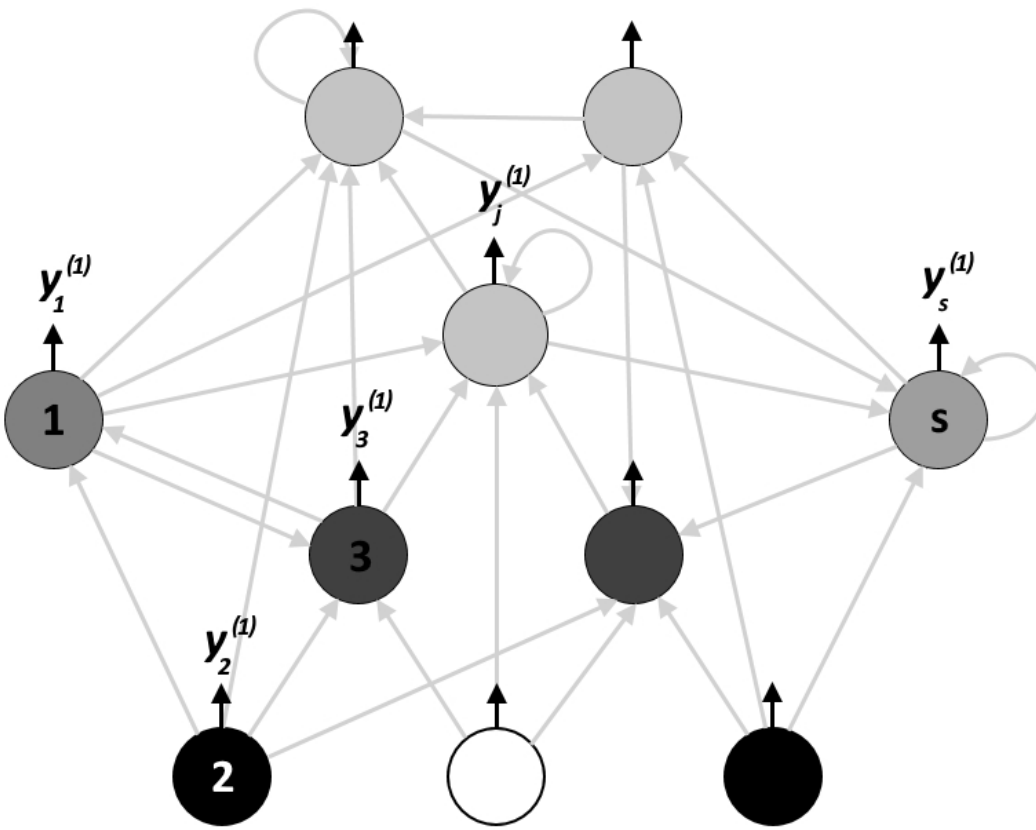
Discrete-Time Computational Dynamics – Initial State

$t = 0$: initial network state $\mathbf{y}^{(0)} \in \{0, 1\}^s$



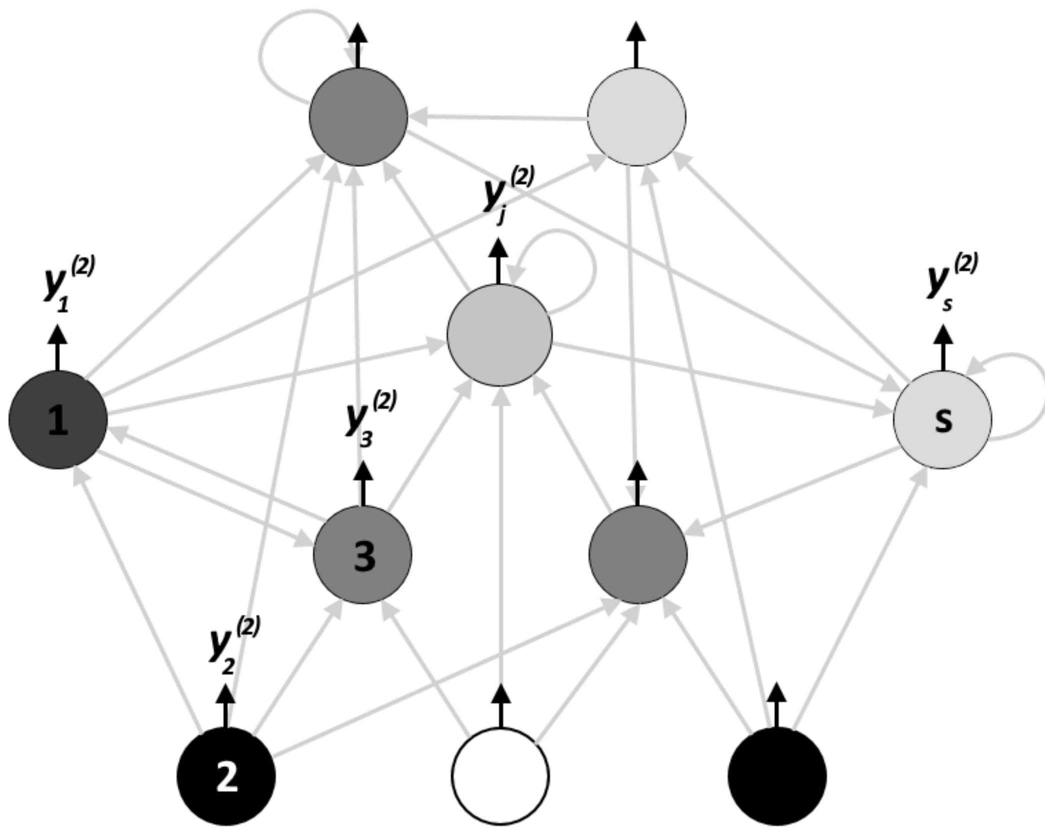
Discrete-Time Computational Dynamics: $t = 1$

$t = 1$: network state $\mathbf{y}^{(1)} \in [0, 1]^s$



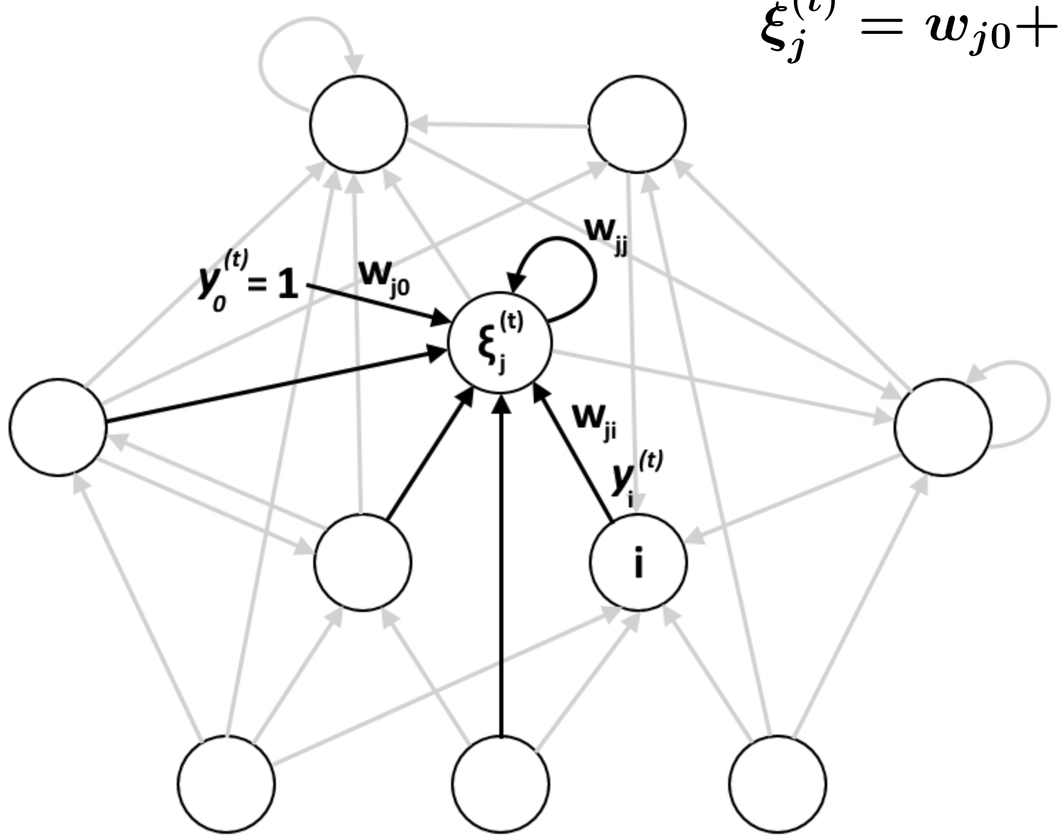
Discrete-Time Computational Dynamics: $t = 2$

$t = 2$: network state $\mathbf{y}^{(2)} \in [0, 1]^s$



Discrete-Time Computational Dynamics – Excitations

at discrete time instant $t \geq 0$, an excitation is computed as



$$\xi_j^{(t)} = w_{j0} + \sum_{i=1}^{s} w_{ji} y_i^{(t)} = \sum_{i=0}^{s} w_{ji} y_i^{(t)}$$

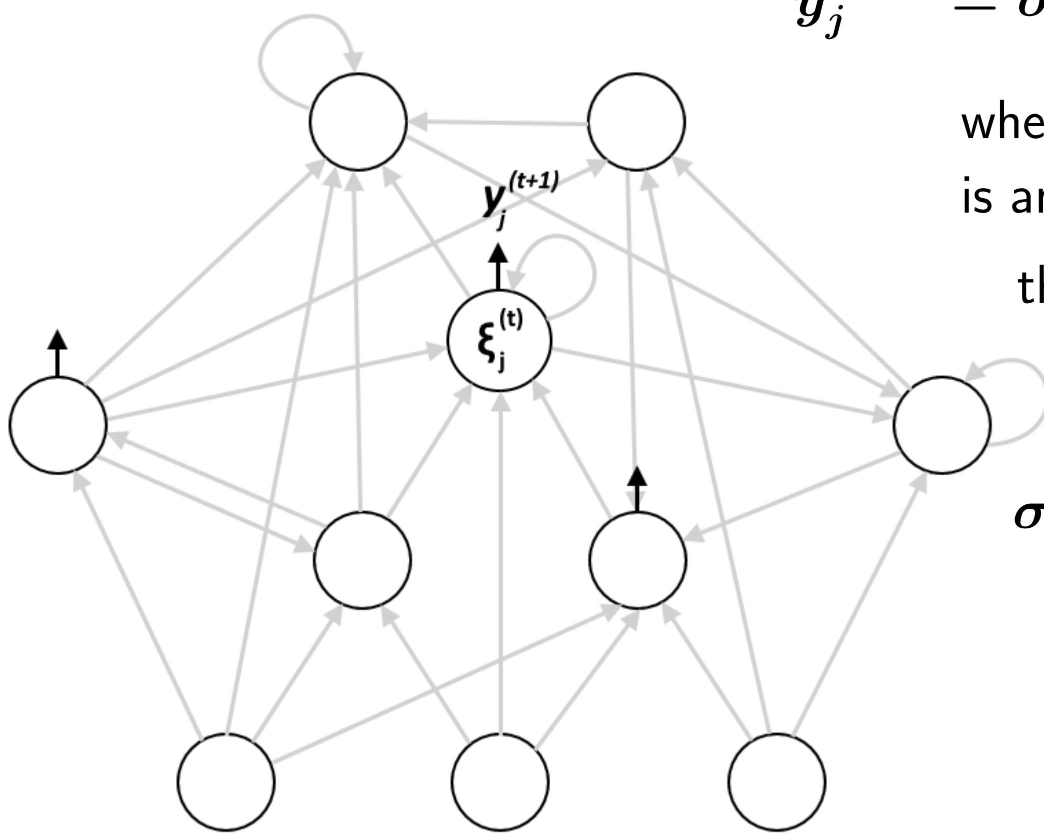
for $j = 1, \ldots, s$

where unit $0 \in V$ has constant output $y_0^{(t)} \equiv 1$ for every $t \geq 0$

Discrete-Time Computational Dynamics – Outputs

at the next time instant $t+1$, every neuron $j \in V$ updates its state:
(fully parallel mode)

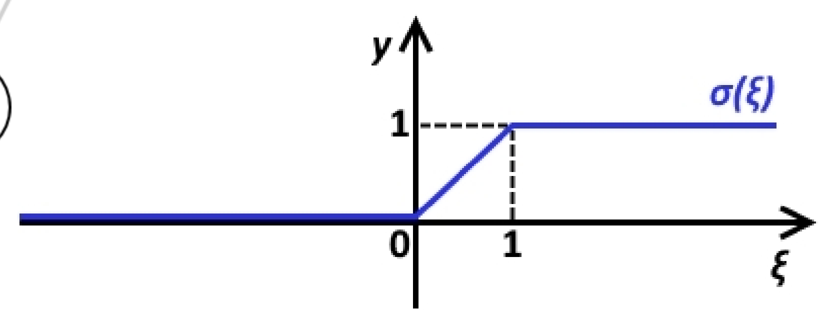
$$y_j^{(t+1)} = \sigma_j\left(\xi_j^{(t)}\right) \quad \text{for } j = 1, \ldots, s$$



where $\sigma_j : \mathbb{R} \longrightarrow [0,1]$
is an activation function, e.g.

the saturated-linear function σ,

$$\sigma(\xi) = \begin{cases} 1 & \text{for } \xi \geq 1 \\ \xi & \text{for } 0 < \xi < 1 \\ 0 & \text{for } \xi \leq 0 \end{cases}$$



The Computational Power of NNs – Motivations

- the potential and limits of general-purpose computation with NNs:

  What is ultimately or efficiently computable by particular NN models?

- idealized mathematical models of practical NNs which abstract away from implementation issues, e.g. analog numerical parameters are true real numbers

- methodology: the computational power and efficiency of NNs is investigated by comparing formal NNs to traditional computational models such as finite automata, Turing machines, Boolean circuits, etc.

- NNs may serve as reference models for analyzing alternative computational resources (other than time or memory space) such as analog state, continuous time, energy, temporal coding, etc.

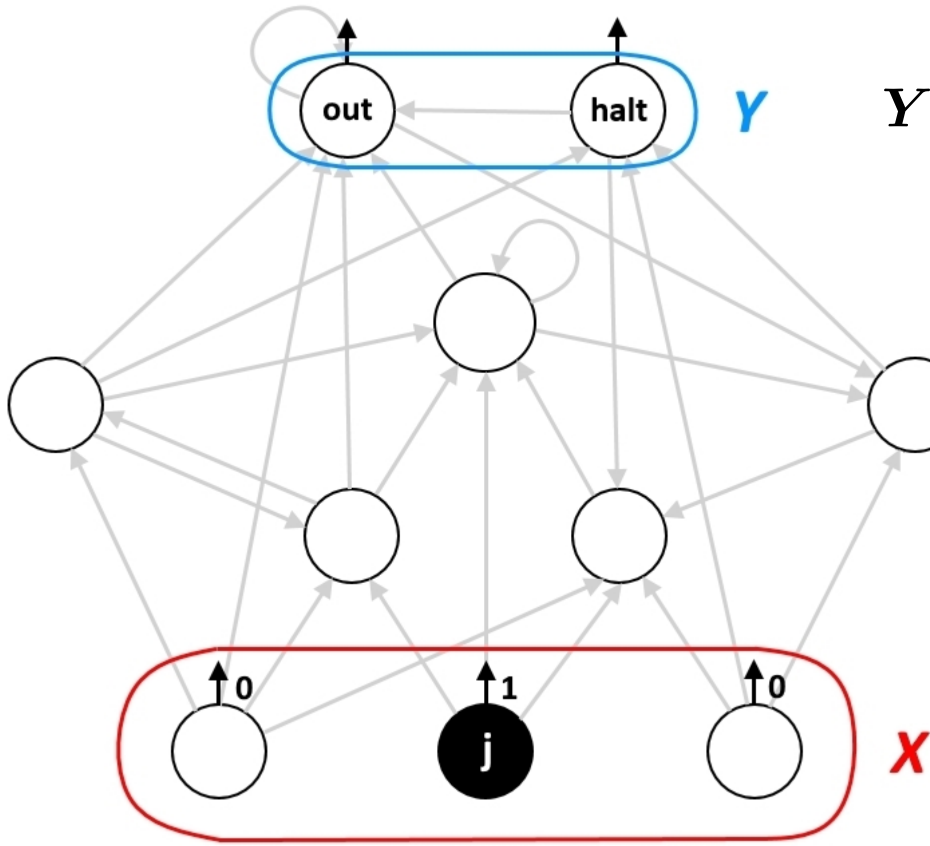
- NNs capture basic characteristics of biological nervous systems (plenty of densely interconnected simple unreliable computational units)

  $\longrightarrow$ computational principles of mental processes

Neural Networks As Formal Language Acceptors

language (problem) $L \subseteq \Sigma^*$ over a finite alphabet Σ

$$y_{\text{out}}^{(T(n))} = \begin{cases} 1 & \text{if } \mathrm{x} \in L \\ 0 & \text{if } \mathrm{x} \notin L \end{cases} \qquad y_{\text{halt}}^{(t)} = \begin{cases} 1 & \text{if } t = T(n) \\ 0 & \text{if } t \neq T(n) \end{cases}$$



$Y = \{\text{out}, \text{halt}\}$ output neurons

$T(n)$ is the computational time in terms of input length $n \geq 0$

online I/O: $T(n) = nd$

$d \geq 1$ is the time overhead for processing a single input symbol

$X = \text{enum}(\Sigma) \subseteq V$ input neurons

$$\uparrow \quad y_j^{(d(i-1))} = 1 \quad \text{iff } j = \text{enum}(x_i)$$

$$\mathrm{x} = x_1 x_2 \ldots x_{i-1} \longleftarrow x_i \longleftarrow x_{i+1} x_{i+2} \ldots x_n \in \Sigma^*$$ input word

The Computational Power of Neural Networks

depends on the information contents of weight parameters:

1. integer weights: finite automaton `(Minsky, 1967)`

2. rational weights: Turing machine `(Siegelmann, Sontag, 1995)`
   polynomial time $\equiv$ complexity class P

3. arbitrary real weights: "super-Turing" computation `(Siegelmann, Sontag, 1994)`
   polynomial time $\equiv$ nonuniform complexity class P/poly
   exponential time $\equiv$ any I/O mapping

The Computational Power of Neural Networks

depends on the information contents of weight parameters:

1. integer weights: finite automaton (Minsky, 1967)

2. rational weights: Turing machine (Siegelmann, Sontag, 1995)
   polynomial time $\equiv$ complexity class P

   polynomial time & increasing Kolmogorov complexity of real weights $\equiv$
   a proper hierarchy of nonuniform complexity classes between P and P/poly
   (Balcázar, Gavaldà, Siegelmann, 1997)

3. arbitrary real weights: "super-Turing" computation (Siegelmann, Sontag, 1994)
   polynomial time $\equiv$ nonuniform complexity class P/poly
   exponential time $\equiv$ any I/O mapping

The Computational Power of Neural Networks

depends on the information contents of weight parameters:

1. integer weights: finite automaton `(Minsky, 1967)`

   a gap between integer a rational weights w.r.t. the Chomsky hierarchy

   regular (Type-3) $\times$ recursively enumerable (Type-0) languages

2. rational weights: Turing machine `(Siegelmann, Sontag, 1995)`
   polynomial time $\equiv$ complexity class P

   polynomial time & increasing Kolmogorov complexity of real weights $\equiv$
   a proper hierarchy of nonuniform complexity classes between P and P/poly
   `(Balcázar, Gavaldà, Siegelmann, 1997)`

3. arbitrary real weights: "super-Turing" computation `(Siegelmann, Sontag, 1994)`
   polynomial time $\equiv$ nonuniform complexity class P/poly
   exponential time $\equiv$ any I/O mapping

Between Integer and Rational Weights

25 neurons with rational weights can implement any Turing machine (Indyk,1995)

?? What is the computational power of a few extra analog neurons ??

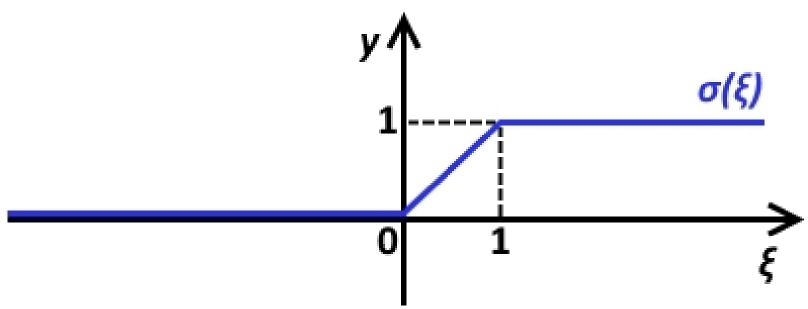
A Neural Network with c Extra Analog Neurons (cANN)

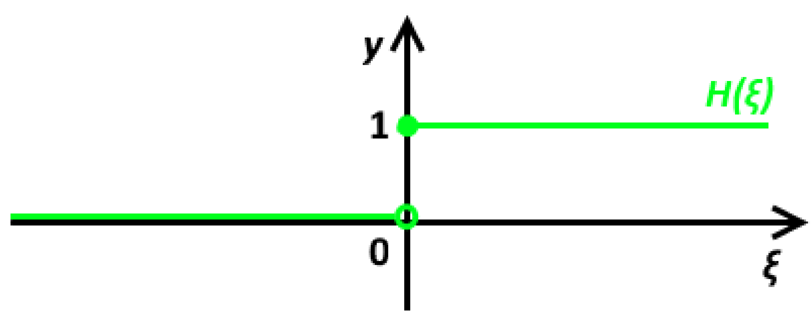
is composed of binary-state neurons with the Heaviside activation function except for the first c analog-state units with the saturated-linear activation function:

$$\sigma_j(\xi) = \begin{cases} \sigma(\xi) = \begin{cases} 1 & \text{for } \xi \geq 1 \\ \xi & \text{for } 0 < \xi < 1 \\ 0 & \text{for } \xi \leq 0 \end{cases} & j = 1, \ldots, c \\ \\ H(\xi) = \begin{cases} 1 & \text{for } \xi \geq 0 \\ 0 & \text{for } \xi < 0 \end{cases} & j = c+1, \ldots, s \end{cases}$$

saturated-linear function

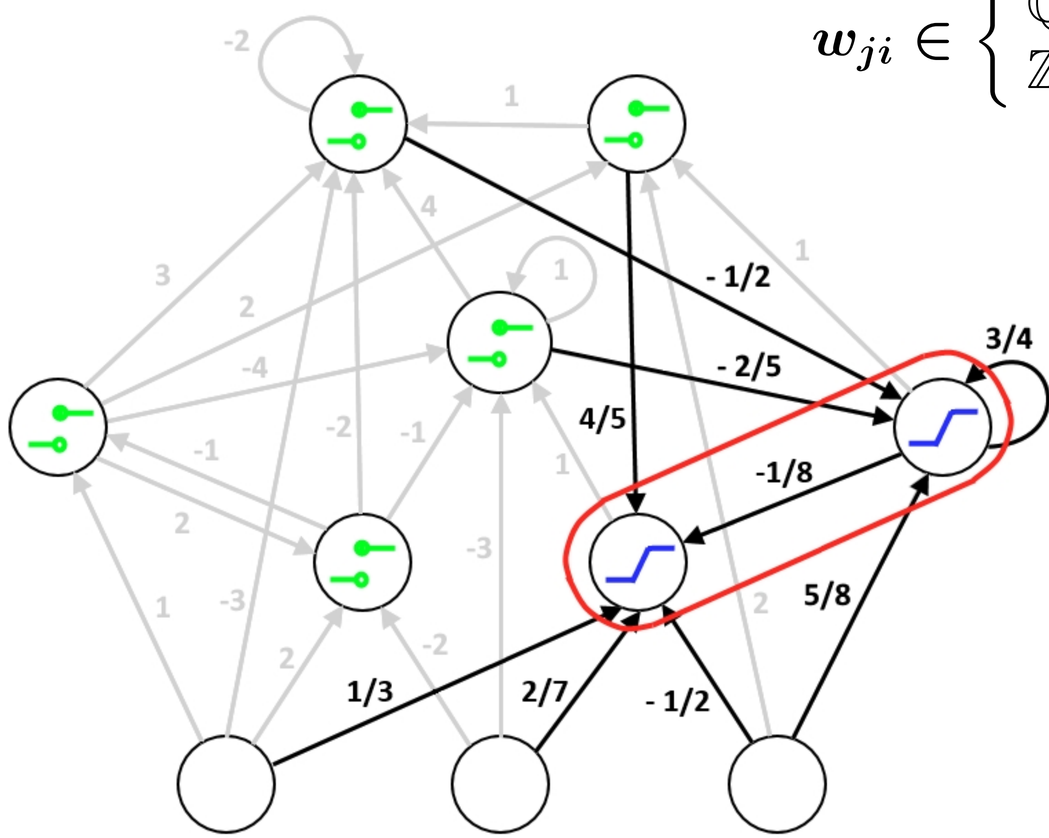
Heaviside function





cANN with Rational Weights

w.l.o.g.: all the weights to neurons are integers except for the first c units with rational weights:



$$w_{ji} \in \begin{cases} \mathbb{Q} & j = 1, \ldots, c \\ \mathbb{Z} & j = c+1, \ldots, s \end{cases}$$

$$i \in \{0, \ldots, s\}$$

1ANNs & the Chomsky Hierarchy

rational-weight NNs $\equiv$ TMs $\equiv$ recursively enumerable languages **(Type-0)**

online 1ANNs $\subset$ LBA $\equiv$ context-sensitive languages **(Type-1)**

1ANNs $\not\subset$ PDA $\equiv$ context-free languages **(Type-2)**

integer-weight NNs $\equiv$ **"quasi-periodic"** 1ANNs $\equiv$ FA $\equiv$ regular languages **(Type-3)**

Non-Standard Positional Numeral Systems

- a real base (radix) β such that $|\beta| > 1$
- a finite set $A \neq \emptyset$ of real digits

β-expansion of a real number $x \in \mathbb{R}$ using the digits from $a_k \in A$, $k \geq 1$:

$$x = (0\,.\,a_1\,a_2\,a_3\;\ldots)_\beta = \sum_{k=1}^{\infty} a_k \beta^{-k}$$

Examples:

- decimal expansions: $\beta = 10$, $A = \{0, 1, 2, \ldots, 9\}$

  $\frac{3}{4} = (0\,.\,74\,\overline{9})_{10} = 7 \cdot 10^{-1} + 5 \cdot 10^{-2} + 9 \cdot 10^{-3} + 9 \cdot 10^{-4} + \cdots$

  any number has at most 2 decimal expansions, e.g. $(0\,.\,74\,\overline{9})_{10} = (0\,.\,75\,\overline{0})_{10}$

- non-integer base: $\beta = \frac{5}{2}$, $A = \left\{0, \frac{1}{2}, \frac{7}{4}\right\}$

  $\frac{3}{4} = \left(0\,.\,\frac{7}{4}\,\frac{1}{2}\,0\,\overline{\frac{7}{4}\,0}\right)_{\frac{5}{2}} = \frac{7}{4} \cdot \left(\frac{5}{2}\right)^{-1} + \frac{1}{2} \cdot \left(\frac{5}{2}\right)^{-2} + 0 \cdot \left(\frac{5}{2}\right)^{-3} + \frac{7}{4} \cdot \left(\frac{5}{2}\right)^{-4} + \cdots$

  most of the representable numbers has a continuum of distinct β-expansions,

  e.g. $\frac{3}{4} = \left(0\,.\,\overline{\frac{7}{4}\,\frac{1}{2}\,\frac{1}{2}\;\ldots\;\frac{1}{2}\,0}\right)_{\frac{5}{2}}$

Quasi-Periodic β-Expansion

eventually periodic β-expansions:

$$\Big(0\,.\,\underbrace{a_1\ \ldots\ a_{m_1}}_{\text{preperiodic part}}\ \underbrace{\overline{a_{m_1+1}\ \ldots\ a_{m_2}}}_{\text{repetend}}\Big)_\beta \qquad \left(\text{e.g. } \frac{19}{55} = (0\,.\,3\,\overline{45})_{10}\right)$$

eventually quasi-periodic β-expansions:

$$\Big(0\,.\,\underbrace{a_1\ldots a_{m_1}}_{\text{preperiodic part}}\ \underbrace{a_{m_1+1}\ldots a_{m_2}}_{\text{quasi-repetend}}\ \underbrace{a_{m_2+1}\ldots a_{m_3}}_{\text{quasi-repetend}}\ \underbrace{a_{m_3+1}\ldots a_{m_4}}_{\text{quasi-repetend}}\ \ldots\Big)_\beta$$

such that

$$(0\,.\,\overline{a_{m_1+1}\ldots a_{m_2}})_\beta = (0\,.\,\overline{a_{m_2+1}\ldots a_{m_3}})_\beta = (0\,.\,\overline{a_{m_3+1}\ldots a_{m_4}})_\beta = \cdots$$

Example: the plastic $\beta \approx 1.324718$ ($\beta^3 - \beta - 1 = 0$), $A = \{0, 1\}$

$$1 = (0\,.\,0\ \underbrace{\underline{100}}\ \underbrace{\underline{0\,011\,011\,1}}\ \underbrace{\underline{0\,011\,1}}\ \underbrace{\underline{100}}\ldots)_\beta$$

with quasi-repetends: $(0\,.\,\overline{100})_\beta = (0\,.\,\overline{0(011)^i1})_\beta = \beta$ for every $i \geq 1$

Quasi-Periodic Numbers

$r \in \mathbb{R}$ is a β-quasi-periodic number within A if every β-expansions of r is eventually quasi-periodic

Examples:

- r from the complement of the Cantor set **is** 3-quasi-periodic within $A = \{0, 2\}$
  (r has no β-expansion at all)

- $r = \frac{3}{4}$ **is** $\frac{5}{2}$-quasi-periodic within $A = \{0\,,\,\frac{1}{2}\,,\,\frac{7}{4}\}$

- $r = 1$ **is** β-quasi-periodic within $A = \{0, 1\}$ for the plastic $\beta \approx 1.324718$

- $r \in \mathbb{Q}(\beta)$ **is** β-quasi-periodic within $A \subset \mathbb{Q}(\beta)$ for Pisot β
  (a real algebraic integer $\beta > 1$ whose all Galois conjugates $\beta' \in \mathbb{C}$ satisfy $|\beta'| < 1$)

- $r = \frac{40}{57} = (0\,.\,0\,\overline{011})_{\frac{3}{2}}$ **is not** $\frac{3}{2}$-quasi-periodic within $A = \{0, 1\}$
  (greedy $\frac{3}{2}$-expansion of $\frac{40}{57} = (0\,.\,100000001\ldots)_{\frac{3}{2}}$ is not eventually periodic)

Regular 1ANNs

Theorem (`Šíma, IJCNN 2017`). *Let $\mathcal{N}$ be a 1ANN such that the feedback weight of its analog neuron satisfies $0 < |w_{11}| < 1$. Denote*

$$\beta = \frac{1}{w_{11}}\,, \quad A = \left\{ \sum_{i \in V \setminus \{1\}} \frac{w_{1i}}{w_{11}} y_i \,\middle|\, y_2, \ldots, y_s \in \{0,1\} \right\} \cup \{0, \beta\}\,,$$

$$R = \left\{ -\sum_{i \in V \setminus \{1\}} \frac{w_{ji}}{w_{j1}} y_i \,\middle|\, j \in V \setminus (X \cup \{1\}) \text{ s.t. } w_{j1} \neq 0\,, \right.$$
$$\left. y_2, \ldots, y_s \in \{0,1\} \right\} \cup \{0,1\}\,.$$

If every $r \in R$ is β-quasi-periodic within A, then $\mathcal{N}$ accepts a regular language.

Corollary. *Let $\mathcal{N}$ be a 1ANN such that $\beta = \frac{1}{w_{11}}$ is a Pisot number whereas all the remaining weights are from $\mathbb{Q}(\beta)$. Then $\mathcal{N}$ accepts a regular language.*

Examples: 1ANNs with rational weights + the feedback weight of analog neuron:

- $w_{11} = 1/n$ for any integer $n \in \mathbb{N}$
- $w_{11} = 1/\beta$ for the plastic constant $\beta = \frac{\sqrt[3]{9-\sqrt{69}} + \sqrt[3]{9+\sqrt{69}}}{\sqrt[3]{18}} \approx 1.324718$
- $w_{11} = 1/\varphi$ for the golden ratio $\varphi = \frac{1+\sqrt{5}}{2} \approx 1.618034$

An Upper Bound on the Number of Analog Neurons

What is the number c of analog neurons to make the cANNs with rational weights Turing-complete (universal) ?? (`Indyk,1995:` $c \leq 25$)

Our main technical result: 3ANNs can simulate any Turing machine

Theorem. *Given a Turing machine $\mathcal{M}$ that accepts a language $L = \mathcal{L}(\mathcal{M})$ in time $T(n)$, there is a 3ANN $\mathcal{N}$ with rational weights, which accepts the same language $L = \mathcal{L}(\mathcal{N})$ in time $O(T(n))$.*

$\longrightarrow$ **refining the analysis of cANNs within the Chomsky Hierarchy:**

rational-weight 3ANNs $\equiv$ TMs $\equiv$ recursively enumerable languages **(Type-0)**

online 1ANNs $\subset$ LBA $\equiv$ context-sensitive languages **(Type-1)**

1ANNs $\not\subset$ PDA $\equiv$ context-free languages **(Type-2)**

integer-weight NNs $\equiv$ "quasi-periodic" 1ANNs $\equiv$ FA $\equiv$ regular languages **(Type-3)**

Idea of Proof – Stack Encoding

Turing machine $\equiv$ 2-stack pushdown automaton (2PDA)

$\longrightarrow$ an analog neuron implements a stack

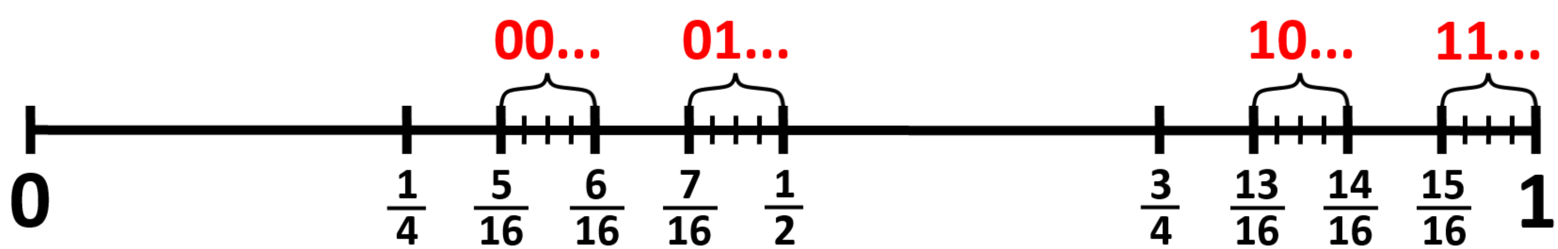
the stack contents $x_1 \ldots x_n \in \{0,1\}^*$ is encoded by an analog state of a neuron using Cantor-like set (Siegelmann, Sontag, 1995):

$$\mathtt{code}(x_1 \ldots x_n) = \sum_{i=1}^{n} \frac{2x_i + 1}{4^i} \in [0,1]$$

that is, $\mathtt{code}(0\, x_2 \ldots x_n) \in \left[\frac{1}{4}, \frac{1}{2}\right)$ vs. $\mathtt{code}(1\, x_2 \ldots x_n) \in \left[\frac{3}{4}, 1\right)$

$\mathtt{code}(00\, x_3 \ldots x_n) \in \left[\frac{5}{16}, \frac{6}{16}\right)$ vs. $\mathtt{code}(01\, x_2 \ldots x_n) \in \left[\frac{7}{16}, \frac{1}{2}\right)$

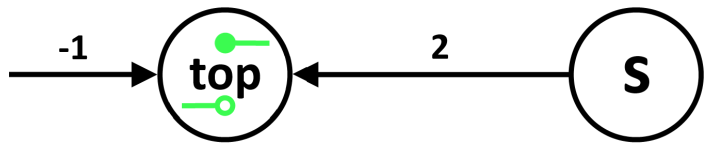
$\mathtt{code}(10\, x_3 \ldots x_n) \in \left[\frac{13}{16}, \frac{14}{16}\right)$ vs. $\mathtt{code}(11\, x_2 \ldots x_n) \in \left[\frac{15}{16}, 1\right)$ etc.



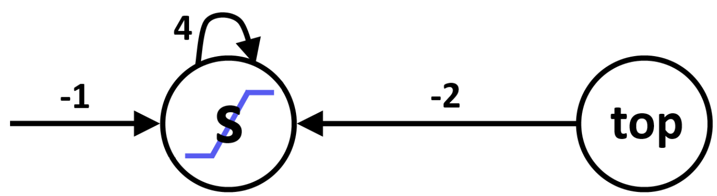
Idea of Proof – Stack Operations

implementing the stack operations on $s = \mathtt{code}(x_1 \dots x_n) \in [0,1]$:

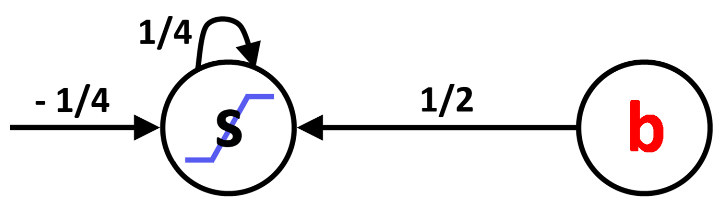
- $\mathtt{top}(s) = H(2s-1) = \begin{cases} 1 & \text{if } s \geq \frac{1}{2} \quad \text{i.e. } s = \mathtt{code}(1\, x_2 \dots x_n) \\ 0 & \text{if } s < \frac{1}{2} \quad \text{i.e. } s = \mathtt{code}(0\, x_2 \dots x_n) \end{cases}$



- $\mathtt{pop}(s) = \sigma(4s - 2\,\mathtt{top}(s) - 1) = \mathtt{code}(x_2 \dots x_n)$



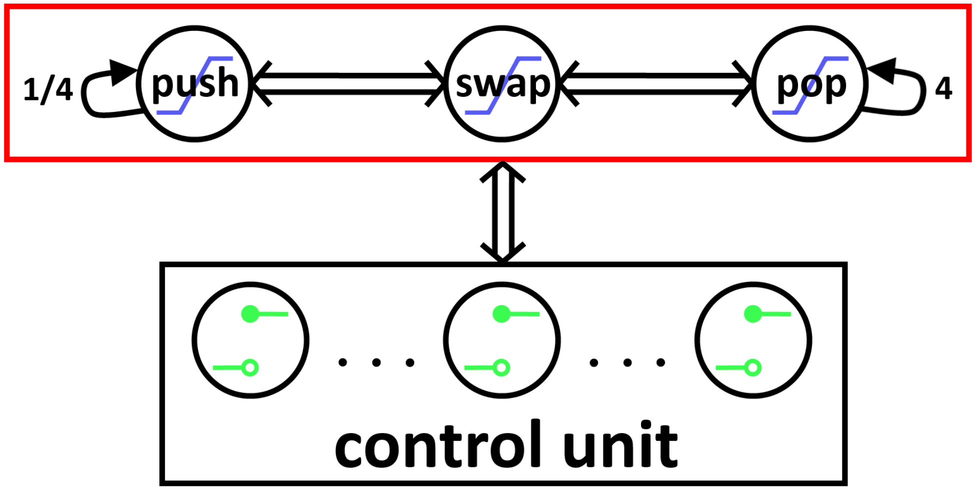
- $\mathtt{push}(s, b) = \sigma\left(\frac{s}{4} + \frac{2b-1}{4}\right) = \mathtt{code}(b\, x_1 \dots x_n)$ for $b \in \{0, 1\}$



Idea of Proof – 2PDA implementation by 3ANN

$\mathbf{2}$ stacks are implemented by $\mathbf{2}$ analog neurons computing $\mathtt{push}$ and $\mathtt{pop}$, respectively

$\longrightarrow$ the 3rd analog neuron of 3ANN performs the $\mathtt{swap}$ operation



$\mathbf{2}$ types of instructions depending on whether the $\mathtt{push}$ and $\mathtt{pop}$ operations apply to the matching neurons:

1. short instruction: $\mathtt{push(b);\ pop}$
2. long instruction: $\mathtt{push(top);\ pop;\ swap;\ push(b);\ pop}$

$+$ a complicated synchronization of the fully parallel 3ANN $\square$

Conclusion & Open Problems

- We have refined the analysis of NNs with rational weights by showing that 3ANNs are Turing-complete.

- Are 1ANNs or 2ANNs Turing-complete?

  conjecture: 1ANNs do not recognize the non-regular context-free languages (CFL$\setminus$REG) vs. CFL$\subset$2ANNs

- a necessary condition for a 1ANN to accept a regular language

- a proper hierarchy of NNs e.g. with increasing quasi-period of weights