Expressive power of Recurrent Neural Networks over Infinite Input Streams

Jérémie Cabessa

Laboratoire d'Économie Mathématique
Université Paris 2

27 May 2016

Introduction

- We assume that some aspects of information processing in the brain can be approached from the perspective of computability theory.
- The computational capabilities of recurrent neural networks have mainly been studied in the context of classical computation (McCulloch & Pitts, Turing, Kleene, von Neumann, Minsky, Papert,..., Siegelmann & Sontag,...).
- Here, we consider recurrent neural networks involved in infinite (or non-terminating) computations, i.e., working on infinite input streams.

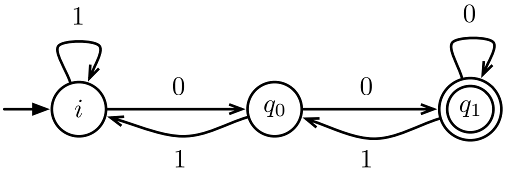
Introduction

- We assume that some aspects of information processing in the brain can be approached from the perspective of computability theory.
- The computational capabilities of recurrent neural networks have mainly been studied in the context of classical computation (McCulloch & Pitts, Turing, Kleene, von Neumann, Minsky, Papert,..., Siegelmann & Sontag,...).
- Here, we consider recurrent neural networks involved in infinite (or non-terminating) computations, i.e., working on infinite input streams.

Introduction

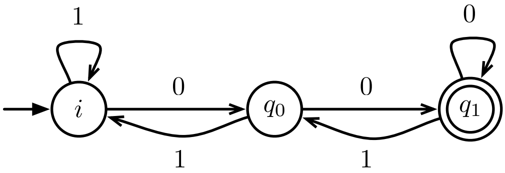
- We assume that some aspects of information processing in the brain can be approached from the perspective of computability theory.
- The computational capabilities of recurrent neural networks have mainly been studied in the context of classical computation (McCulloch & Pitts, Turing, Kleene, von Neumann, Minsky, Papert,..., Siegelmann & Sontag,...).
- Here, we consider recurrent neural networks involved in infinite (or non-terminating) computations, i.e., working on infinite input streams.

Finite State Automaton



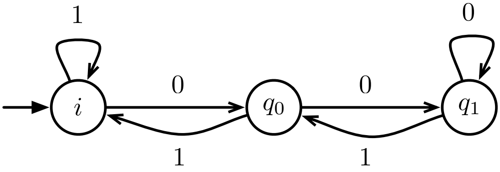
- an input u is *accepted* by $\mathcal{A}$ if $\mathcal{A}(u)$ reaches a final state
- an input u is *rejected* by $\mathcal{A}$ otherwise

Finite State Automaton



- an input u is *accepted* by $\mathcal{A}$ if $\mathcal{A}(u)$ reaches a final state
- an input u is *rejected* by $\mathcal{A}$ otherwise

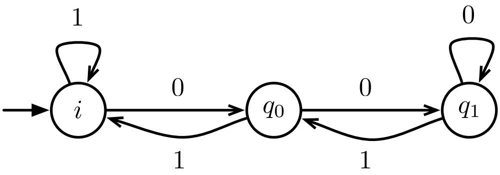
Muller Automaton



$$\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$$

- an input u is *accepted* by $\mathcal{A}$ if $\inf(\rho_u) \in \mathcal{T}$
- an input u is *rejected* by $\mathcal{A}$ otherwise

Muller Automaton

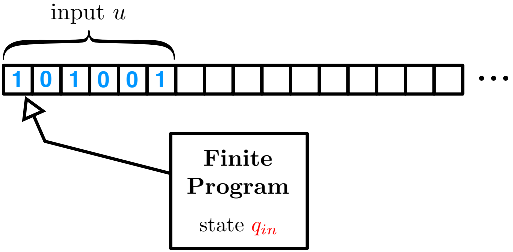


$$\mathcal{T} = \{\{q_0, q_1\}, \{q_1\}\}$$

- an input u is *accepted* by $\mathcal{A}$ if $\inf(\rho_u) \in \mathcal{T}$
- an input u is *rejected* by $\mathcal{A}$ otherwise

Turing Machine

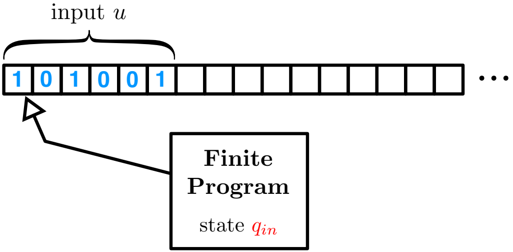
A *Turing machine* (TM) consists of an infinite tape, a read-write head, and a finite program.



- input u is *accepted* by $\mathcal{M}$ if $\mathcal{M}(u)$ reaches the state q_{acc}
- input u is *rejected* by $\mathcal{M}$ if $\mathcal{M}(u)$ reaches the state q_{rej}

Turing Machine

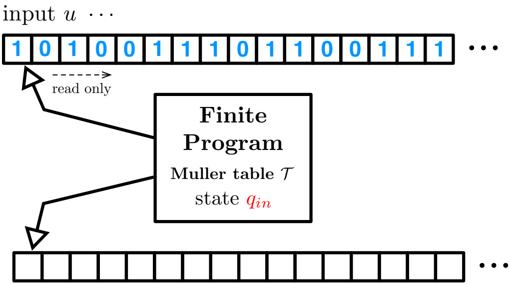
A *Turing machine* (TM) consists of an infinite tape, a read-write head, and a finite program.



- input u is *accepted* by $\mathcal{M}$ if $\mathcal{M}(u)$ reaches the state q_{acc}
- input u is *rejected* by $\mathcal{M}$ if $\mathcal{M}(u)$ reaches the state q_{rej}

Muller Turing Machine

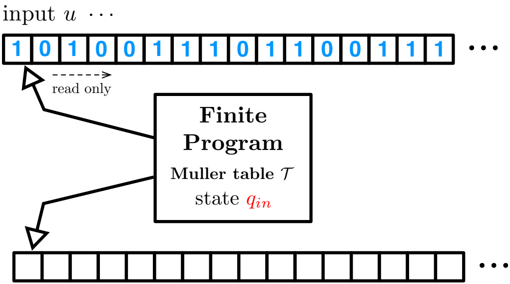
A *Muller Turing machine* consists of a classical TM with Muller acceptance condition.



- input u is *accepted* by $\mathcal{M}$ if $\inf(\rho_u) \in \mathcal{T}$
- input u is *rejected* by $\mathcal{M}$ otherwise

Muller Turing Machine

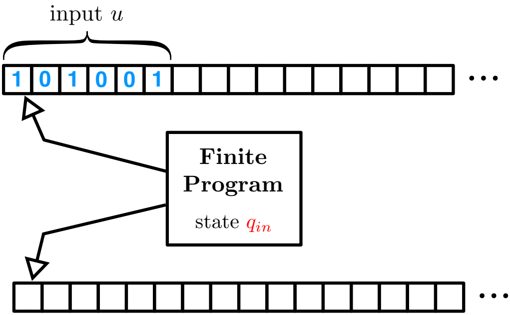
A *Muller Turing machine* consists of a classical TM with Muller acceptance condition.



- input u is *accepted* by $\mathcal{M}$ if $\inf(\rho_u) \in \mathcal{T}$
- input u is *rejected* by $\mathcal{M}$ otherwise

Turing Machine with Advice

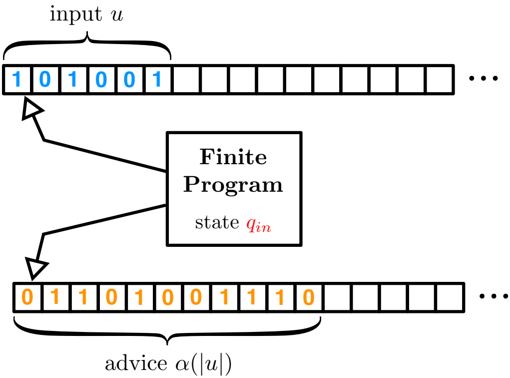
A *Turing machine with advice* (TM/A) is a TM provided with an additional advice tape and advice function $\alpha : \mathbb{N} \longrightarrow \{0, 1\}^*$.



- **P/poly** is the class of languages recognized in polynomial time by Turing machines with polynomial advices (TM/poly(A)).

Turing Machine with Advice

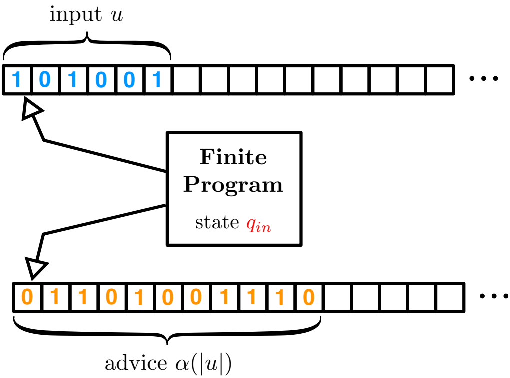
A *Turing machine with advice* (TM/A) is a TM provided with an additional advice tape and advice function $\alpha : \mathbb{N} \longrightarrow \{0,1\}^*$.



- **P/poly** is the class of languages recognized in polynomial time by Turing machines with polynomial advices (TM/poly(A)).

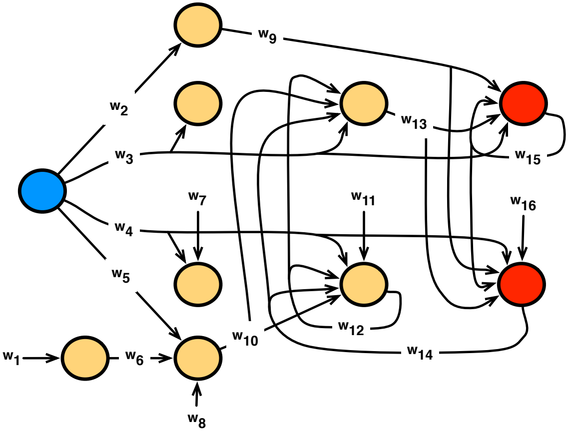
Turing Machine with Advice

A *Turing machine with advice* (TM/A) is a TM provided with an additional advice tape and advice function $\alpha : \mathbb{N} \longrightarrow \{0,1\}^*$.

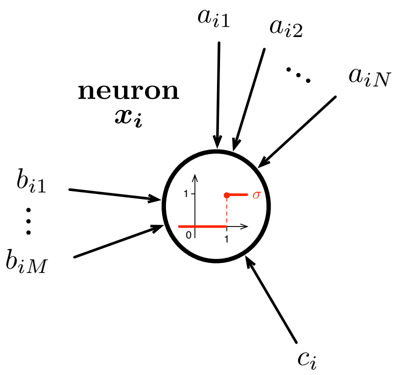


- **P/poly** is the class of languages recognized in polynomial time by Turing machines with polynomial advices (TM/poly(A)).

Recurrent Neural Network

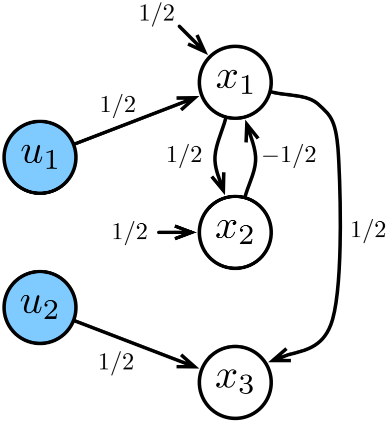


Boolean Recurrent Neural Networks

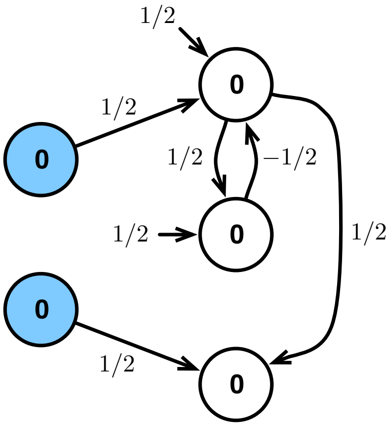


$$x_i(t+1) = \theta\left(\sum_{j=1}^{N} a_{ij} \cdot x_j(t) + \sum_{j=1}^{M} b_{ij} \cdot u_j(t) + c_i\right)$$

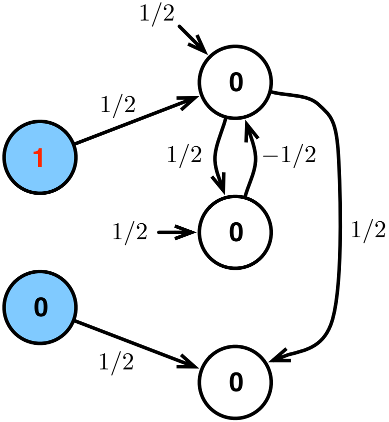
Boolean Recurrent Neural Networks



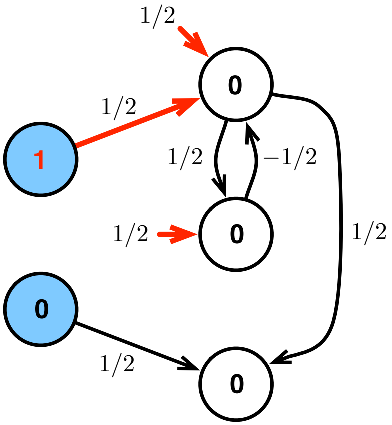
Boolean Recurrent Neural Networks



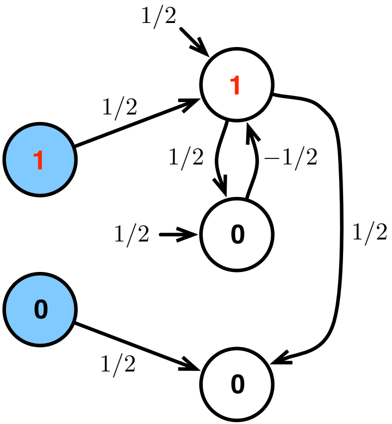
Boolean Recurrent Neural Networks



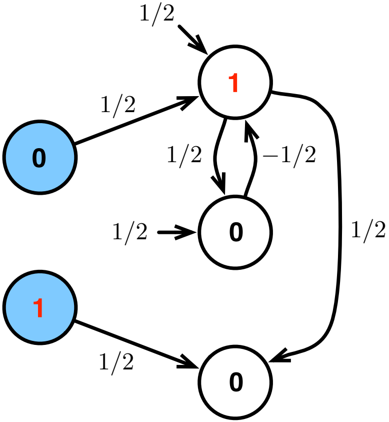
Boolean Recurrent Neural Networks



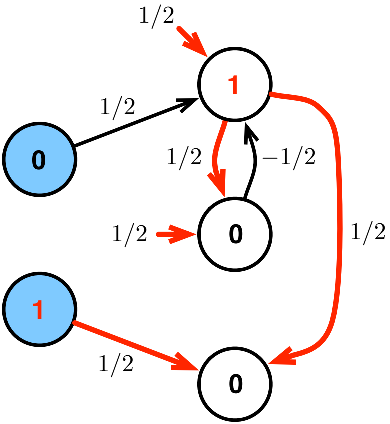
Boolean Recurrent Neural Networks



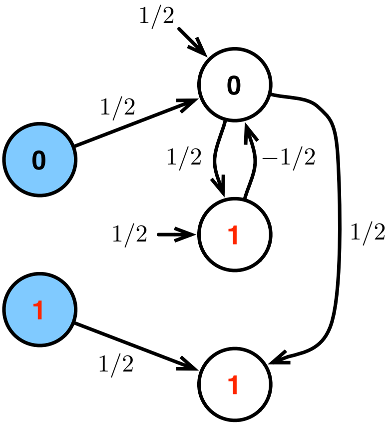
Boolean Recurrent Neural Networks



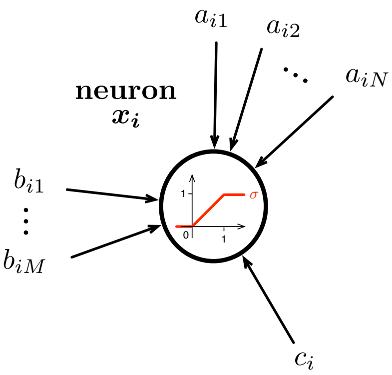
Boolean Recurrent Neural Networks



Boolean Recurrent Neural Networks

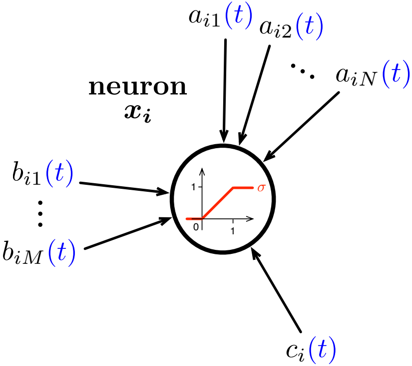


Sigmoidal Recurrent Neural Networks



$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij} \cdot x_j(t) + \sum_{j=1}^{M} b_{ij} \cdot u_j(t) + c_i\right)$$

Sigmoidal Recurrent Neural Networks



$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t) \cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t) \cdot u_j(t) + c_i(t)\right)$$

Recurrent Neural Networks

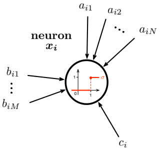
We consider eight models of RNNs:

1. Boolean rational RNNs: B-RNN[$\mathbb{Q}$]s
2. Boolean real RNNs: B-RNN[$\mathbb{R}$]s
3. Sigmoidal static rational RNNs: St-RNN[$\mathbb{Q}$]s
4. Sigmoidal static real RNNs: St-RNN[$\mathbb{R}$]s
5. Sigmoidal bi-valued evolving rational RNNs: Ev$_2$-RNN[$\mathbb{Q}$]s
6. Sigmoidal bi-valued evolving real RNNs: Ev$_2$-RNN[$\mathbb{R}$]s
7. Sigmoidal general evolving rational RNNs: Ev-RNN[$\mathbb{Q}$]s
8. Sigmoidal general evolving real N-RNNs: Ev-RNN[$\mathbb{R}$]s

Recurrent Neural Networks

We consider eight models of RNNs:

1. Boolean rational RNNs: B-RNN[$\mathbb{Q}$]s
2. Boolean real RNNs: B-RNN[$\mathbb{R}$]s
3. Sigmoidal static rational RNNs: St-RNN[$\mathbb{Q}$]s
4. Sigmoidal static real RNNs: St-RNN[$\mathbb{R}$]s
5. Sigmoidal bi-valued evolving rational RNNs: Ev_2-RNN[$\mathbb{Q}$]s
6. Sigmoidal bi-valued evolving real RNNs: Ev_2-RNN[$\mathbb{R}$]s
7. Sigmoidal general evolving rational RNNs: Ev-RNN[$\mathbb{Q}$]s
8. Sigmoidal general evolving real N-RNNs: Ev-RNN[$\mathbb{R}$]s

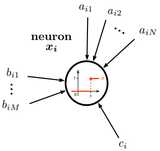


$$x_i(t+1) = \theta\left(\sum_{j=1}^{N} a_{ij} \cdot x_j(t) + \sum_{j=1}^{M} b_{ij} \cdot u_j(t) + c_i\right)$$

Recurrent Neural Networks

We consider eight models of RNNs:

1. Boolean rational RNNs: B-RNN[$\mathbb{Q}$]s
2. Boolean real RNNs: B-RNN[$\mathbb{R}$]s
3. Sigmoidal static rational RNNs: St-RNN[$\mathbb{Q}$]s
4. Sigmoidal static real RNNs: St-RNN[$\mathbb{R}$]s
5. Sigmoidal bi-valued evolving rational RNNs: Ev$_2$-RNN[$\mathbb{Q}$]s
6. Sigmoidal bi-valued evolving real RNNs: Ev$_2$-RNN[$\mathbb{R}$]s
7. Sigmoidal general evolving rational RNNs: Ev-RNN[$\mathbb{Q}$]s
8. Sigmoidal general evolving real N-RNNs: Ev-RNN[$\mathbb{R}$]s

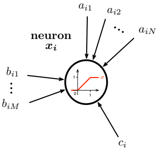


$$x_i(t+1) = \theta\left(\sum_{j=1}^{N} a_{ij} \cdot x_j(t) + \sum_{j=1}^{M} b_{ij} \cdot u_j(t) + c_i\right)$$

Recurrent Neural Networks

We consider eight models of RNNs:

1. Boolean rational RNNs: B-RNN[$\mathbb{Q}$]s
2. Boolean real RNNs: B-RNN[$\mathbb{R}$]s
3. Sigmoidal static rational RNNs: St-RNN[$\mathbb{Q}$]s
4. Sigmoidal static real RNNs: St-RNN[$\mathbb{R}$]s
5. Sigmoidal bi-valued evolving rational RNNs: Ev_2-RNN[$\mathbb{Q}$]s
6. Sigmoidal bi-valued evolving real RNNs: Ev_2-RNN[$\mathbb{R}$]s
7. Sigmoidal general evolving rational RNNs: Ev-RNN[$\mathbb{Q}$]s
8. Sigmoidal general evolving real N-RNNs: Ev-RNN[$\mathbb{R}$]s

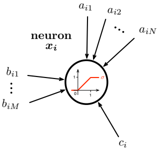


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij} \cdot x_j(t) + \sum_{j=1}^{M} b_{ij} \cdot u_j(t) + c_i\right)$$

Recurrent Neural Networks

We consider eight models of RNNs:

1. Boolean rational RNNs: B-RNN[$\mathbb{Q}$]s
2. Boolean real RNNs: B-RNN[$\mathbb{R}$]s
3. Sigmoidal static rational RNNs: St-RNN[$\mathbb{Q}$]s
4. Sigmoidal static real RNNs: St-RNN[$\mathbb{R}$]s
5. Sigmoidal bi-valued evolving rational RNNs: Ev_2-RNN[$\mathbb{Q}$]s
6. Sigmoidal bi-valued evolving real RNNs: Ev_2-RNN[$\mathbb{R}$]s
7. Sigmoidal general evolving rational RNNs: Ev-RNN[$\mathbb{Q}$]s
8. Sigmoidal general evolving real N-RNNs: Ev-RNN[$\mathbb{R}$]s

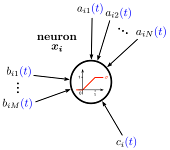


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij} \cdot x_j(t) + \sum_{j=1}^{M} b_{ij} \cdot u_j(t) + c_i\right)$$

Recurrent Neural Networks

We consider eight models of RNNs:

1. Boolean rational RNNs: B-RNN[$\mathbb{Q}$]s
2. Boolean real RNNs: B-RNN[$\mathbb{R}$]s
3. Sigmoidal static rational RNNs: St-RNN[$\mathbb{Q}$]s
4. Sigmoidal static real RNNs: St-RNN[$\mathbb{R}$]s
5. Sigmoidal bi-valued evolving rational RNNs: Ev$_2$-RNN[$\mathbb{Q}$]s
6. Sigmoidal bi-valued evolving real RNNs: Ev$_2$-RNN[$\mathbb{R}$]s
7. Sigmoidal general evolving rational RNNs: Ev-RNN[$\mathbb{Q}$]s
8. Sigmoidal general evolving real N-RNNs: Ev-RNN[$\mathbb{R}$]s

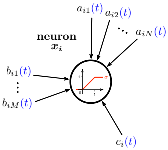


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t)\cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t)\cdot u_j(t) + c_i(t)\right)$$

Recurrent Neural Networks

We consider eight models of RNNs:

1. Boolean rational RNNs: B-RNN[$\mathbb{Q}$]s
2. Boolean real RNNs: B-RNN[$\mathbb{R}$]s
3. Sigmoidal static rational RNNs: St-RNN[$\mathbb{Q}$]s
4. Sigmoidal static real RNNs: St-RNN[$\mathbb{R}$]s
5. Sigmoidal bi-valued evolving rational RNNs: Ev$_2$-RNN[$\mathbb{Q}$]s
6. Sigmoidal bi-valued evolving real RNNs: Ev$_2$-RNN[$\mathbb{R}$]s
7. Sigmoidal general evolving rational RNNs: Ev-RNN[$\mathbb{Q}$]s
8. Sigmoidal general evolving real N-RNNs: Ev-RNN[$\mathbb{R}$]s

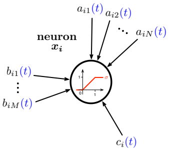


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t) \cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t) \cdot u_j(t) + c_i(t)\right)$$

Recurrent Neural Networks

We consider eight models of RNNs:

1. Boolean rational RNNs: B-RNN[$\mathbb{Q}$]s
2. Boolean real RNNs: B-RNN[$\mathbb{R}$]s
3. Sigmoidal static rational RNNs: St-RNN[$\mathbb{Q}$]s
4. Sigmoidal static real RNNs: St-RNN[$\mathbb{R}$]s
5. Sigmoidal bi-valued evolving rational RNNs: Ev_2-RNN[$\mathbb{Q}$]s
6. Sigmoidal bi-valued evolving real RNNs: Ev_2-RNN[$\mathbb{R}$]s
7. Sigmoidal general evolving rational RNNs: Ev-RNN[$\mathbb{Q}$]s
8. Sigmoidal general evolving real N-RNNs: Ev-RNN[$\mathbb{R}$]s

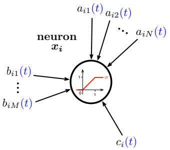


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t) \cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t) \cdot u_j(t) + c_i(t)\right)$$

Recurrent Neural Networks

We consider eight models of RNNs:

1. Boolean rational RNNs: B-RNN[$\mathbb{Q}$]s
2. Boolean real RNNs: B-RNN[$\mathbb{R}$]s
3. Sigmoidal static rational RNNs: St-RNN[$\mathbb{Q}$]s
4. Sigmoidal static real RNNs: St-RNN[$\mathbb{R}$]s
5. Sigmoidal bi-valued evolving rational RNNs: Ev$_2$-RNN[$\mathbb{Q}$]s
6. Sigmoidal bi-valued evolving real RNNs: Ev$_2$-RNN[$\mathbb{R}$]s
7. Sigmoidal general evolving rational RNNs: Ev-RNN[$\mathbb{Q}$]s
8. Sigmoidal general evolving real N-RNNs: Ev-RNN[$\mathbb{R}$]s



$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t)\cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t)\cdot u_j(t) + c_i(t)\right)$$

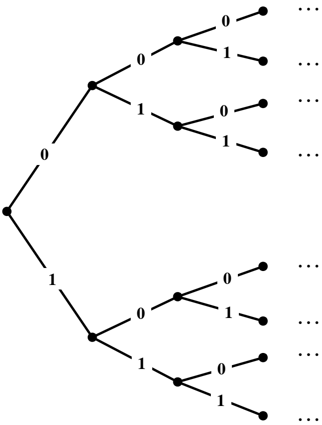
Results (Classical Computation)

	Boolean	Static	Bi-valued Evolving	Evolving
$\mathbb{Q}$	FSA **REG** Kl 56, Mi 67	TM **P** S&S 95	TM/poly(A) **P/poly** C&S 11,14	TM/poly(A) **P/poly** C&S 11,14
$\mathbb{R}$	FSA **REG** Kl 56, Mi 67	TM/poly(A) **P/poly** S&S 94	TM/poly(A) **P/poly** C&S 11,14	TM/poly(A) **P/poly** C&S 11,14

Topology

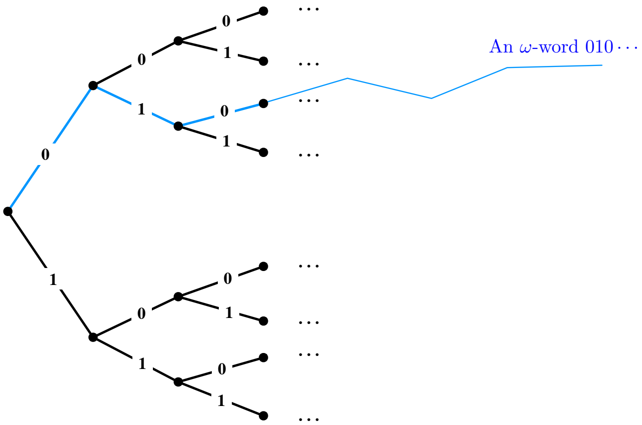
The Cantor space $\{0,1\}^\omega$

the set of infinite sequences of bits



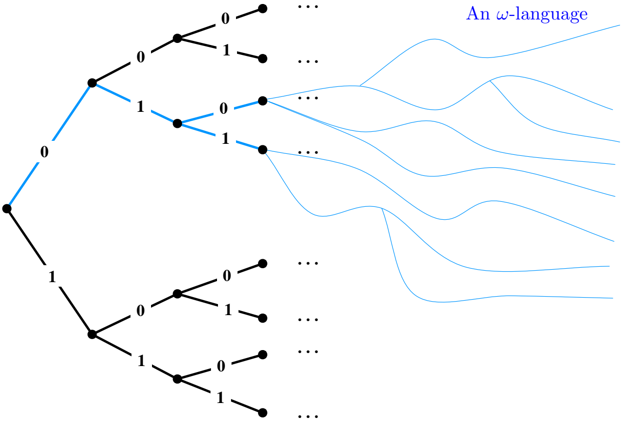
Topology

The Cantor space $\{0,1\}^\omega$
the set of infinite sequences of bits



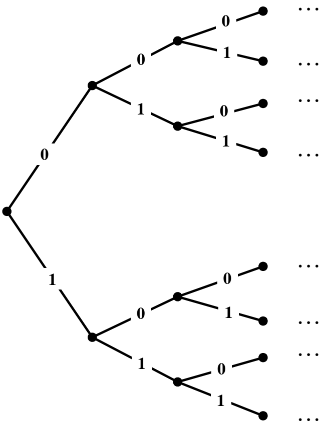
Topology

The Cantor space $\{0,1\}^\omega$
the set of infinite sequences of bits



Topology

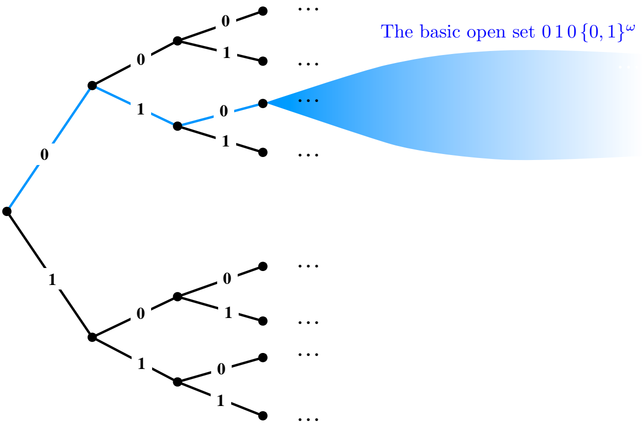
The Cantor space $\{0,1\}^\omega$
the set of infinite sequences of bits



equipped with product topology
of the discrete topology on $\{0,1\}$

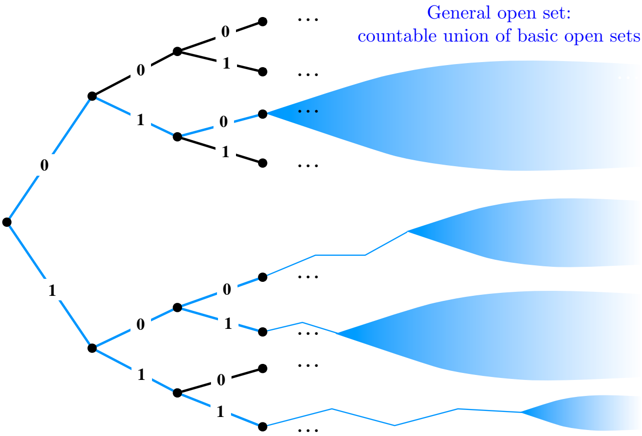
Topology

The Cantor space $\{0,1\}^\omega$
the set of infinite sequences of bits



Topology

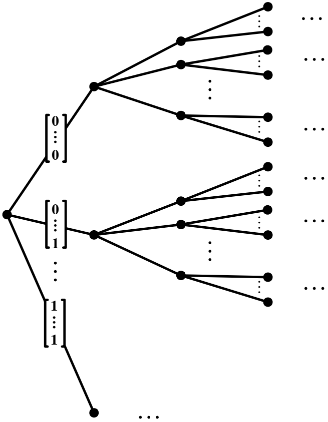
The Cantor space $\{0,1\}^\omega$
the set of infinite sequences of bits



Topology

The space $(\mathbb{B}^N)^\omega$

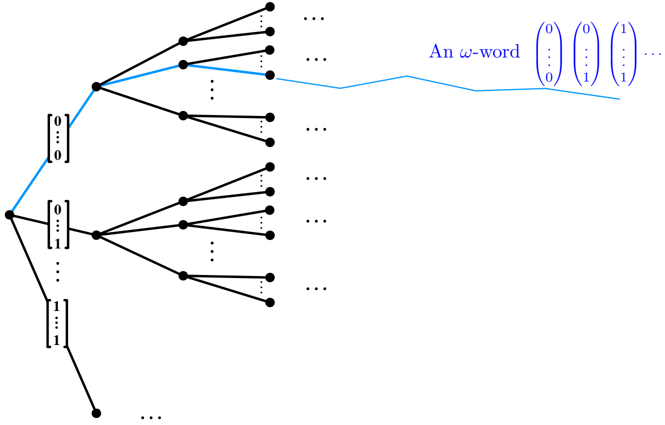
the set of infinite sequences of N-dim. Boolean vectors



Topology

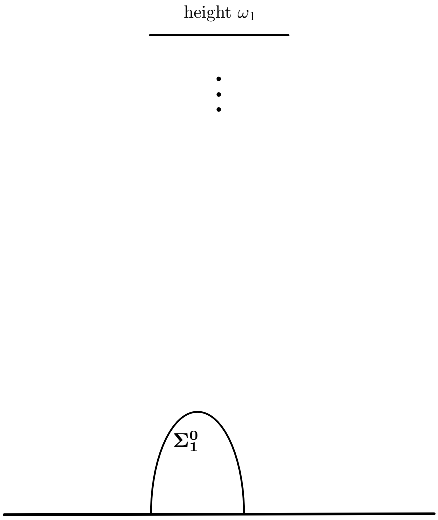
The space $(\mathbb{B}^N)^\omega$

the set of infinite sequences of N-dim. Boolean vectors

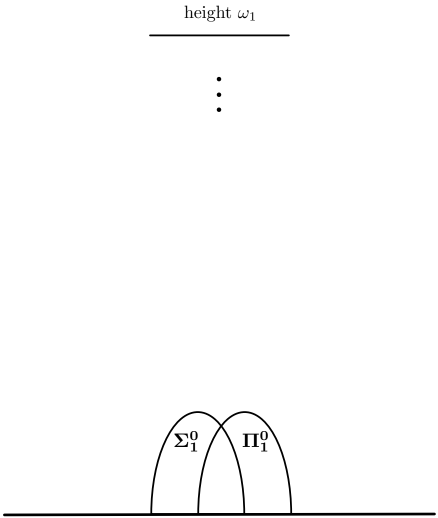


Topology

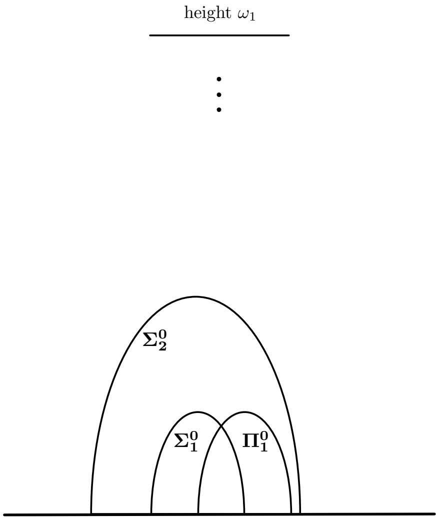
Topology



Topology

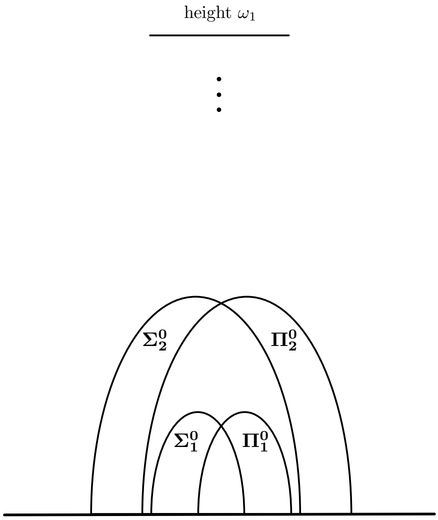


Topology

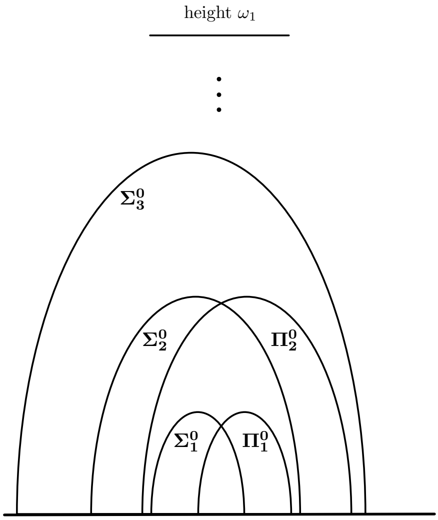


height ω_1
$\vdots$
$\boldsymbol{\Sigma}^0_2$
$\boldsymbol{\Sigma}^0_1$
$\boldsymbol{\Pi}^0_1$

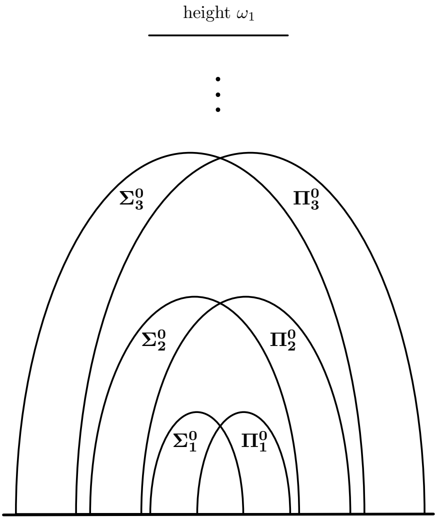
Topology


height ω_1
$\vdots$
$\mathbf{\Sigma^0_2}$
$\mathbf{\Pi^0_2}$
$\mathbf{\Sigma^0_1}$
$\mathbf{\Pi^0_1}$

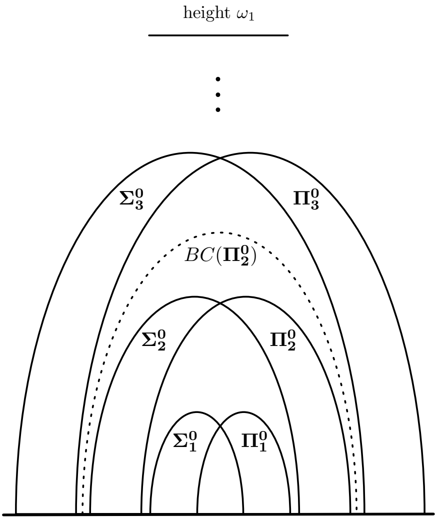
Topology


height ω_1
$\boldsymbol{\Sigma^0_3}$
$\boldsymbol{\Sigma^0_2}$
$\boldsymbol{\Pi^0_2}$
$\boldsymbol{\Sigma^0_1}$
$\boldsymbol{\Pi^0_1}$

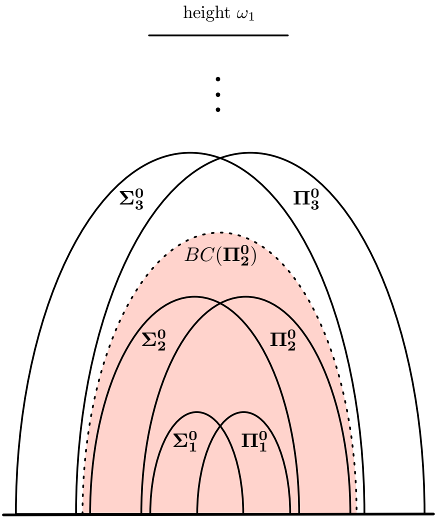
Topology


height ω_1
$\mathbf{\Sigma^0_3}$
$\mathbf{\Pi^0_3}$
$\mathbf{\Sigma^0_2}$
$\mathbf{\Pi^0_2}$
$\mathbf{\Sigma^0_1}$
$\mathbf{\Pi^0_1}$

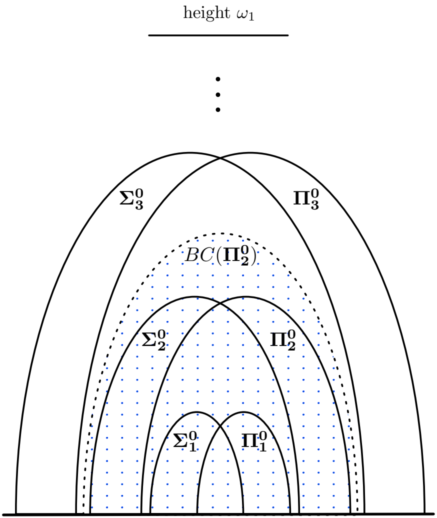
Topology


height ω_1
$\mathbf{\Sigma}^0_3$
$\mathbf{\Pi}^0_3$
$BC(\mathbf{\Pi}^0_2)$
$\mathbf{\Sigma}^0_2$
$\mathbf{\Pi}^0_2$
$\mathbf{\Sigma}^0_1$
$\mathbf{\Pi}^0_1$

Topology


height ω_1
$\boldsymbol{\Sigma}^0_3$
$\boldsymbol{\Pi}^0_3$
$BC(\boldsymbol{\Pi}^0_2)$
$\boldsymbol{\Sigma}^0_2$
$\boldsymbol{\Pi}^0_2$
$\boldsymbol{\Sigma}^0_1$
$\boldsymbol{\Pi}^0_1$

Topology

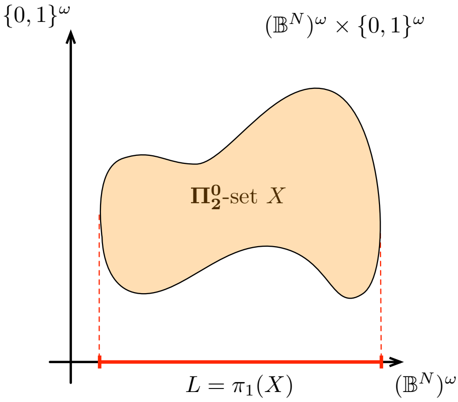

height ω_1
$\boldsymbol{\Sigma}^0_3$
$\boldsymbol{\Pi}^0_3$
$BC(\boldsymbol{\Pi}^0_2)$
$\boldsymbol{\Sigma}^0_2$
$\boldsymbol{\Pi}^0_2$
$\boldsymbol{\Sigma}^0_1$
$\boldsymbol{\Pi}^0_1$

Topology

$$
\begin{tikzcd}[column sep=small]
& \vdots & \\
\boldsymbol{\Sigma^0_3} & & \boldsymbol{\Pi^0_3} \\
& \boldsymbol{\Delta^0_3} \arrow[ul, phantom, sloped, "\subsetneq"] \arrow[ur, phantom, sloped, "\subsetneq"] & \\
\boldsymbol{\Sigma^0_2} \arrow[ur, phantom, sloped, "\subsetneq"] & & \boldsymbol{\Pi^0_2} \arrow[ul, phantom, sloped, "\supsetneq"] \\
& \boldsymbol{\Delta^0_2} \arrow[ul, phantom, sloped, "\subsetneq"] \arrow[ur, phantom, sloped, "\subsetneq"] & \\
\boldsymbol{\Sigma^0_1} \arrow[ur, phantom, sloped, "\subsetneq"] & & \boldsymbol{\Pi^0_1} \arrow[ul, phantom, sloped, "\supsetneq"] \\
& \boldsymbol{\Delta^0_1} \arrow[ul, phantom, sloped, "\subsetneq"] \arrow[ur, phantom, sloped, "\subsetneq"] &
\end{tikzcd}
$$

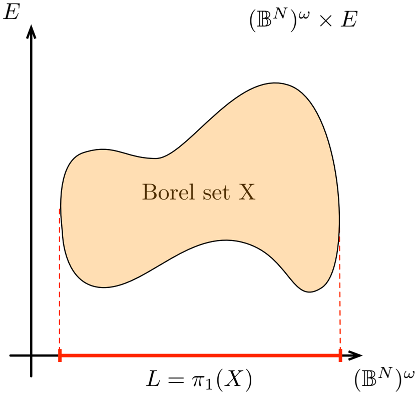
Topology

- An ω-language $L \subseteq (\mathbb{B}^N)^\omega$ is *analytic* ($\mathbf{\Sigma}^1_1$) iff it is the first projection of some $\mathbf{\Pi}^0_2$-set $X \subseteq (\mathbb{B}^N)^\omega \times \{0,1\}^\omega$.



Topology

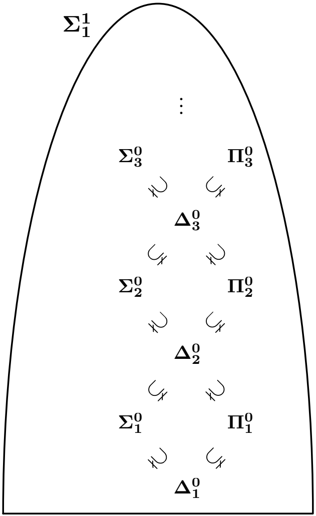
- An ω-language $L \subseteq (\mathbb{B}^N)^\omega$ is *analytic* ($\mathbf{\Sigma}_1^1$) iff it is the first projection of some Borel set $X \subseteq (\mathbb{B}^N)^\omega \times E$, where E is a Polish space.



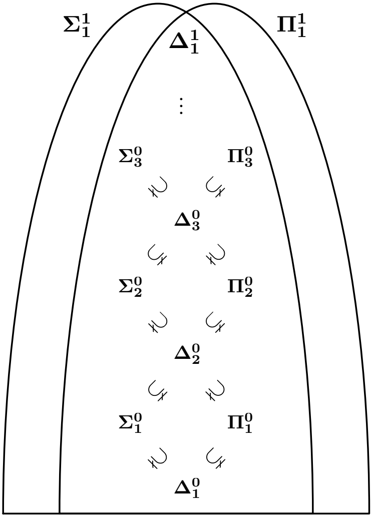
Topology

$$
\begin{tikzcd}
 & \vdots & \\
\boldsymbol{\Sigma}^0_3 & & \boldsymbol{\Pi}^0_3 \\
 & \boldsymbol{\Delta}^0_3 \arrow[ul, hook, "/" marking] \arrow[ur, hook, "/" marking] & \\
\boldsymbol{\Sigma}^0_2 \arrow[ur, hook, "/" marking] & & \boldsymbol{\Pi}^0_2 \arrow[ul, hook, "/" marking] \\
 & \boldsymbol{\Delta}^0_2 \arrow[ul, hook, "/" marking] \arrow[ur, hook, "/" marking] & \\
\boldsymbol{\Sigma}^0_1 \arrow[ur, hook, "/" marking] & & \boldsymbol{\Pi}^0_1 \arrow[ul, hook, "/" marking] \\
 & \boldsymbol{\Delta}^0_1 \arrow[ul, hook, "/" marking] \arrow[ur, hook, "/" marking] &
\end{tikzcd}
$$

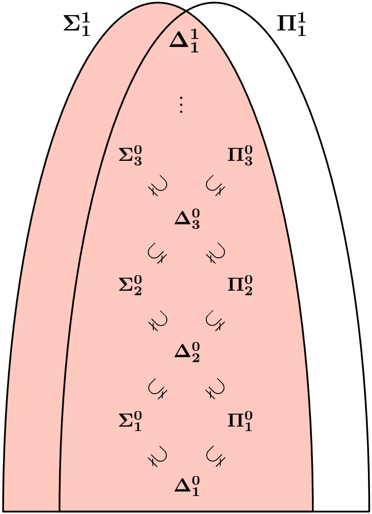
Topology


$\boldsymbol{\Sigma}^1_1$
$\vdots$
$\boldsymbol{\Sigma}^0_3$ $\boldsymbol{\Pi}^0_3$
$\boldsymbol{\Delta}^0_3$
$\boldsymbol{\Sigma}^0_2$ $\boldsymbol{\Pi}^0_2$
$\boldsymbol{\Delta}^0_2$
$\boldsymbol{\Sigma}^0_1$ $\boldsymbol{\Pi}^0_1$
$\boldsymbol{\Delta}^0_1$

Topology

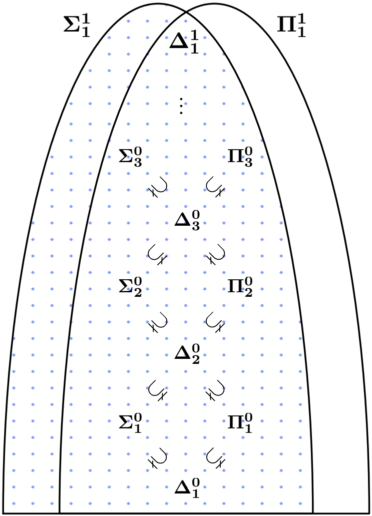

$\mathbf{\Sigma}^1_1$
$\mathbf{\Pi}^1_1$
$\mathbf{\Delta}^1_1$
$\vdots$
$\mathbf{\Sigma}^0_3$
$\mathbf{\Pi}^0_3$
$\mathbf{\Delta}^0_3$
$\mathbf{\Sigma}^0_2$
$\mathbf{\Pi}^0_2$
$\mathbf{\Delta}^0_2$
$\mathbf{\Sigma}^0_1$
$\mathbf{\Pi}^0_1$
$\mathbf{\Delta}^0_1$

Topology



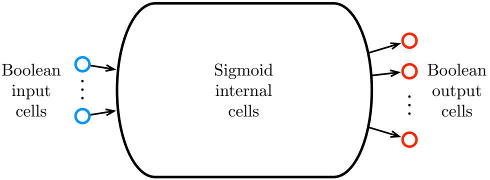
$\boldsymbol{\Sigma}^1_1$
$\boldsymbol{\Pi}^1_1$
$\boldsymbol{\Delta}^1_1$
$\vdots$
$\boldsymbol{\Sigma}^0_3$
$\boldsymbol{\Pi}^0_3$
$\boldsymbol{\Delta}^0_3$
$\boldsymbol{\Sigma}^0_2$
$\boldsymbol{\Pi}^0_2$
$\boldsymbol{\Delta}^0_2$
$\boldsymbol{\Sigma}^0_1$
$\boldsymbol{\Pi}^0_1$
$\boldsymbol{\Delta}^0_1$

Topology


$\mathbf{\Sigma}_1^1$
$\mathbf{\Pi}_1^1$
$\mathbf{\Delta}_1^1$
$\vdots$
$\mathbf{\Sigma}_3^0$
$\mathbf{\Pi}_3^0$
$\mathbf{\Delta}_3^0$
$\mathbf{\Sigma}_2^0$
$\mathbf{\Pi}_2^0$
$\mathbf{\Delta}_2^0$
$\mathbf{\Sigma}_1^0$
$\mathbf{\Pi}_1^0$
$\mathbf{\Delta}_1^0$

Deterministic ω-RNNs

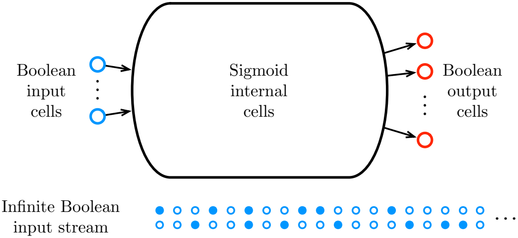
We consider RNNs with Boolean input and output cells, sigmoidal internal cells, and working on infinite input streams.



- Input stream $s \in (\mathbb{B}^M)^\omega$ *accepted* by $\mathcal{N}$ iff $\mathcal{N}(s)$ enters a meaningful attractor.
- Input stream $s \in (\mathbb{B}^M)^\omega$ rejected by $\mathcal{N}$ otherwise.

Deterministic ω-RNNs

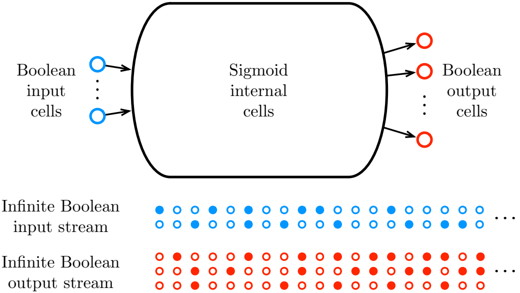
We consider RNNs with Boolean input and output cells, sigmoidal internal cells, and working on infinite input streams.



- Input stream $s \in (\mathbb{B}^M)^\omega$ *accepted* by $\mathcal{N}$ iff $\mathcal{N}(s)$ enters a meaningful attractor.
- Input stream $s \in (\mathbb{B}^M)^\omega$ *rejected* by $\mathcal{N}$ otherwise.

Deterministic ω-RNNs

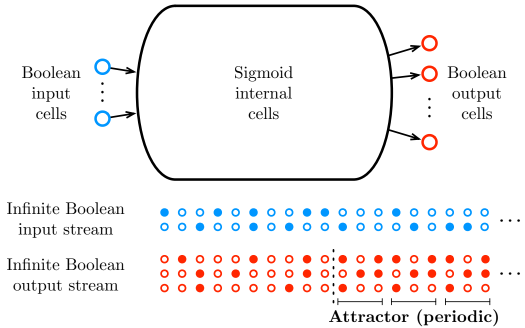
We consider RNNs with Boolean input and output cells, sigmoidal internal cells, and working on infinite input streams.



- Input stream $s \in (\mathbb{B}^M)^\omega$ *accepted* by $\mathcal{N}$ iff $\mathcal{N}(s)$ enters a meaningful attractor.
- Input stream $s \in (\mathbb{B}^M)^\omega$ *rejected* by $\mathcal{N}$ otherwise.

Deterministic ω-RNNs

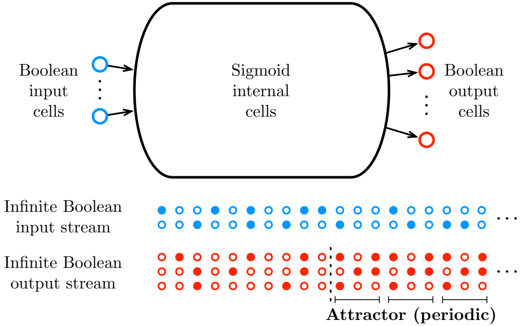
We consider RNNs with Boolean input and output cells, sigmoidal internal cells, and working on infinite input streams.



- Input stream $s \in (\mathbb{B}^M)^\omega$ *accepted* by $\mathcal{N}$ iff $\mathcal{N}(s)$ enters a meaningful attractor.
- Input stream $s \in (\mathbb{B}^M)^\omega$ *rejected* by $\mathcal{N}$ otherwise.

Deterministic ω-RNNs

We consider RNNs with Boolean input and output cells, sigmoidal internal cells, and working on infinite input streams.



- Input stream $s \in (\mathbb{B}^M)^\omega$ *accepted* by $\mathcal{N}$ iff $\mathcal{N}(s)$ enters a meaningful attractor.
- Input stream $s \in (\mathbb{B}^M)^\omega$ *rejected* by $\mathcal{N}$ otherwise.

Deterministic ω-RNNs

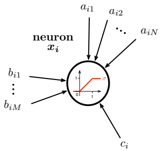
We consider six models of deterministic RNNs:

1. static rational RNNs: D-St-RNN[$\mathbb{Q}$]s
2. static real RNNs: D-St-RNN[$\mathbb{R}$]s
3. bi-valued evolving rational RNNs: D-Ev$_2$-RNN[$\mathbb{Q}$]s
4. bi-valued evolving real RNNs: D-Ev$_2$-RNN[$\mathbb{R}$]s
5. general evolving rational RNNs: D-Ev-RNN[$\mathbb{Q}$]s
6. general evolving real N-RNNs: D-Ev-RNN[$\mathbb{R}$]s

Deterministic ω-RNNs

We consider six models of deterministic RNNs:

1. static rational RNNs: D-St-RNN[$\mathbb{Q}$]s
2. static real RNNs: D-St-RNN[$\mathbb{R}$]s
3. bi-valued evolving rational RNNs: D-Ev$_2$-RNN[$\mathbb{Q}$]s
4. bi-valued evolving real RNNs: D-Ev$_2$-RNN[$\mathbb{R}$]s
5. general evolving rational RNNs: D-Ev-RNN[$\mathbb{Q}$]s
6. general evolving real N-RNNs: D-Ev-RNN[$\mathbb{R}$]s

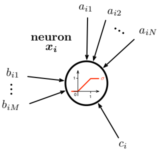


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}\cdot x_j(t) + \sum_{j=1}^{M} b_{ij}\cdot u_j(t) + c_i\right)$$

Deterministic ω-RNNs

We consider six models of deterministic RNNs:

1. static rational RNNs: D-St-RNN[$\mathbb{Q}$]s
2. static real RNNs: D-St-RNN[$\mathbb{R}$]s
3. bi-valued evolving rational RNNs: D-Ev$_2$-RNN[$\mathbb{Q}$]s
4. bi-valued evolving real RNNs: D-Ev$_2$-RNN[$\mathbb{R}$]s
5. general evolving rational RNNs: D-Ev-RNN[$\mathbb{Q}$]s
6. general evolving real N-RNNs: D-Ev-RNN[$\mathbb{R}$]s

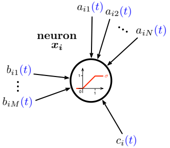


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij} \cdot x_j(t) + \sum_{j=1}^{M} b_{ij} \cdot u_j(t) + c_i\right)$$

Deterministic ω-RNNs

We consider six models of deterministic RNNs:

1. static rational RNNs: D-St-RNN[$\mathbb{Q}$]s
2. static real RNNs: D-St-RNN[$\mathbb{R}$]s
3. bi-valued evolving rational RNNs: D-Ev$_2$-RNN[$\mathbb{Q}$]s
4. bi-valued evolving real RNNs: D-Ev$_2$-RNN[$\mathbb{R}$]s
5. general evolving rational RNNs: D-Ev-RNN[$\mathbb{Q}$]s
6. general evolving real N-RNNs: D-Ev-RNN[$\mathbb{R}$]s

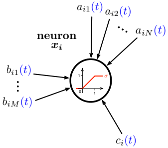


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t) \cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t) \cdot u_j(t) + c_i(t)\right)$$

Deterministic ω-RNNs

We consider six models of deterministic RNNs:

1. static rational RNNs: D-St-RNN[$\mathbb{Q}$]s
2. static real RNNs: D-St-RNN[$\mathbb{R}$]s
3. bi-valued evolving rational RNNs: D-Ev$_2$-RNN[$\mathbb{Q}$]s
4. bi-valued evolving real RNNs: D-Ev$_2$-RNN[$\mathbb{R}$]s
5. general evolving rational RNNs: D-Ev-RNN[$\mathbb{Q}$]s
6. general evolving real N-RNNs: D-Ev-RNN[$\mathbb{R}$]s

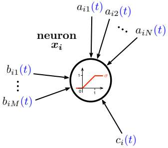


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t) \cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t) \cdot u_j(t) + c_i(t)\right)$$

Deterministic ω-RNNs

We consider six models of deterministic RNNs:

1. static rational RNNs: D-St-RNN[$\mathbb{Q}$]s
2. static real RNNs: D-St-RNN[$\mathbb{R}$]s
3. bi-valued evolving rational RNNs: D-Ev$_2$-RNN[$\mathbb{Q}$]s
4. bi-valued evolving real RNNs: D-Ev$_2$-RNN[$\mathbb{R}$]s
5. general evolving rational RNNs: D-Ev-RNN[$\mathbb{Q}$]s
6. general evolving real N-RNNs: D-Ev-RNN[$\mathbb{R}$]s

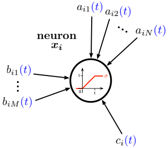


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t)\cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t)\cdot u_j(t) + c_i(t)\right)$$

Deterministic ω-RNNs

We consider six models of deterministic RNNs:

1. static rational RNNs: D-St-RNN[$\mathbb{Q}$]s
2. static real RNNs: D-St-RNN[$\mathbb{R}$]s
3. bi-valued evolving rational RNNs: D-Ev$_2$-RNN[$\mathbb{Q}$]s
4. bi-valued evolving real RNNs: D-Ev$_2$-RNN[$\mathbb{R}$]s
5. general evolving rational RNNs: D-Ev-RNN[$\mathbb{Q}$]s
6. general evolving real N-RNNs: D-Ev-RNN[$\mathbb{R}$]s



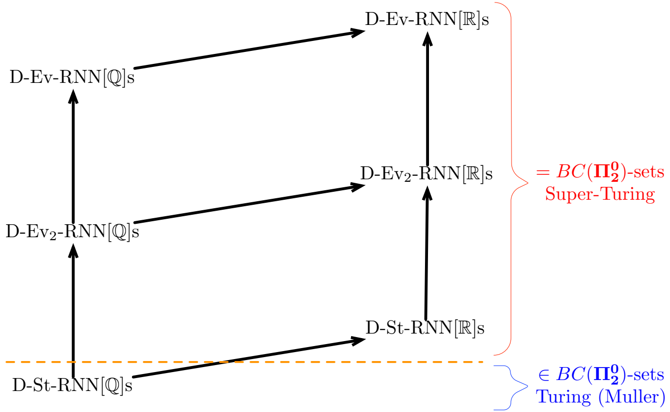
$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t)\cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t)\cdot u_j(t) + c_i(t)\right)$$

Deterministic ω-RNNs

One has the following relationships between those models:

$$\begin{tikzcd}
 & \text{D-Ev-RNN}[\mathbb{R}]\text{s} \\
\text{D-Ev-RNN}[\mathbb{Q}]\text{s} \arrow[ur] & \\
 & \text{D-Ev}_2\text{-RNN}[\mathbb{R}]\text{s} \arrow[uu] \\
\text{D-Ev}_2\text{-RNN}[\mathbb{Q}]\text{s} \arrow[uu] \arrow[ur] & \\
 & \text{D-St-RNN}[\mathbb{R}]\text{s} \arrow[uu] \\
\text{D-St-RNN}[\mathbb{Q}]\text{s} \arrow[uu] \arrow[ur] &
\end{tikzcd}$$

Results


D-Ev-RNN[$\mathbb{R}$]s
D-Ev-RNN[$\mathbb{Q}$]s
D-Ev$_2$-RNN[$\mathbb{R}$]s
D-Ev$_2$-RNN[$\mathbb{Q}$]s
D-St-RNN[$\mathbb{R}$]s
D-St-RNN[$\mathbb{Q}$]s
$= BC(\mathbf{\Pi^0_2})$-sets
Super-Turing
$\in BC(\mathbf{\Pi^0_2})$-sets
Turing (Muller)

Results

Theorem

Let $L \subseteq (\mathbb{B}^M)^\omega$. The following conditions are equivalent.

- *L is recognizable by some deterministic Muller TM (and thus $L \in BC(\mathbf{\Pi^0_2})$)*
- *L is recognizable by some D-St-RNN[$\mathbb{Q}$]*

Proof (sketch): Generalization of the classical equivalence between TMs and St-RNN[$\mathbb{Q}$] (Siegelmann & Sontag 95).

Results

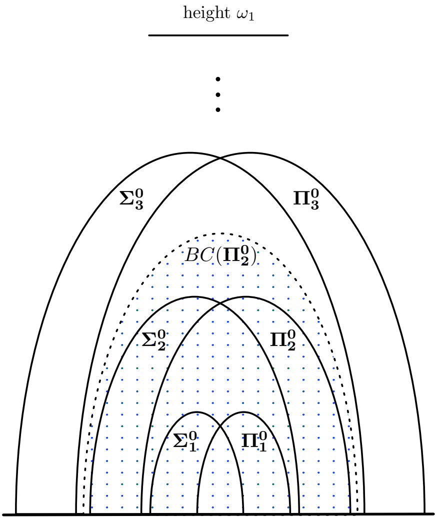
Theorem

Let $L \subseteq (\mathbb{B}^M)^\omega$. The following conditions are equivalent.

- *L is recognizable by some deterministic Muller TM (and thus $L \in BC(\mathbf{\Pi^0_2})$)*
- *L is recognizable by some D-St-RNN[$\mathbb{Q}$]*

Proof (sketch): Generalization of the classical equivalence between TMs and St-RNN[$\mathbb{Q}$] (Siegelmann & Sontag 95).

Results


height ω_1
$\vdots$
$\mathbf{\Sigma^0_3}$
$\mathbf{\Pi^0_3}$
$BC(\mathbf{\Pi^0_2})$
$\mathbf{\Sigma^0_2}$
$\mathbf{\Pi^0_2}$
$\mathbf{\Sigma^0_1}$
$\mathbf{\Pi^0_1}$

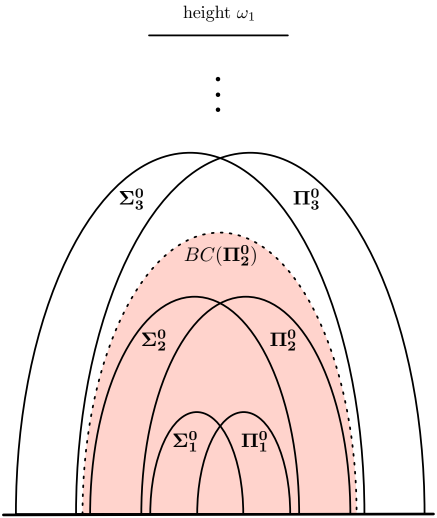
Results

Theorem

Let $L \subseteq (\mathbb{B}^M)^\omega$. The following conditions are equivalent.

- $L \in BC(\mathbf{\Pi^0_2})$;
- L *is recognizable by some D-St-RNN[$\mathbb{R}$];*
- L *is recognizable by some D-Ev$_2$-RNN[$\mathbb{Q}$];*
- L *is recognizable by some D-Ev-RNN[$\mathbb{Q}$];*
- L *is recognizable by some D-Ev$_2$-RNN[$\mathbb{R}$];*
- L *is recognizable by some D-Ev-RNN[$\mathbb{R}$].*

Results



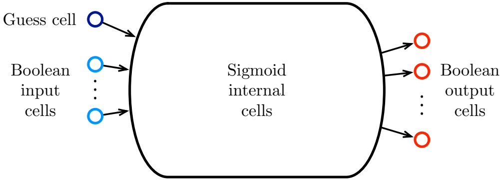
height ω_1
$\vdots$
$\mathbf{\Sigma^0_3}$
$\mathbf{\Pi^0_3}$
$BC(\mathbf{\Pi^0_2})$
$\mathbf{\Sigma^0_2}$
$\mathbf{\Pi^0_2}$
$\mathbf{\Sigma^0_1}$
$\mathbf{\Pi^0_1}$

Results – Summary

DET.	Static	Bi-valued Evolving	General Evolving
$\mathbb{Q}$	D-St-RNN[$\mathbb{Q}$]s $\in BC(\mathbf{\Pi}^0_2)$ Turing (Muller)	D-Ev$_2$-RNN[$\mathbb{Q}$]s $= BC(\mathbf{\Pi}^0_2)$ super-Turing	D-Ev-RNN[$\mathbb{Q}$]s $= BC(\mathbf{\Pi}^0_2)$ super-Turing
$\mathbb{R}$	D-St-RNN[$\mathbb{R}$]s $= BC(\mathbf{\Pi}^0_2)$ super-Turing	D-Ev$_2$-RNN[$\mathbb{R}$]s $= BC(\mathbf{\Pi}^0_2)$ super-Turing	D-Ev-RNN[$\mathbb{R}$]s $= BC(\mathbf{\Pi}^0_2)$ super-Turing

Nondeterminism of Type I

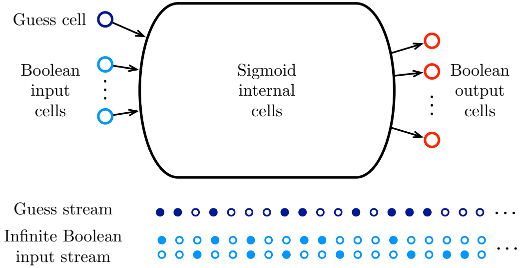
The RNNs are provided with an additional Boolean guess cell.



- Input stream $s \in (\mathbb{B}^M)^\omega$ *accepted* by $\mathcal{N}$ iff there exists some guess $g \in \mathbb{B}^\omega$ s.t. $\mathcal{N}(s,g)$ enters a meaningful attractor.

Nondeterminism of Type I

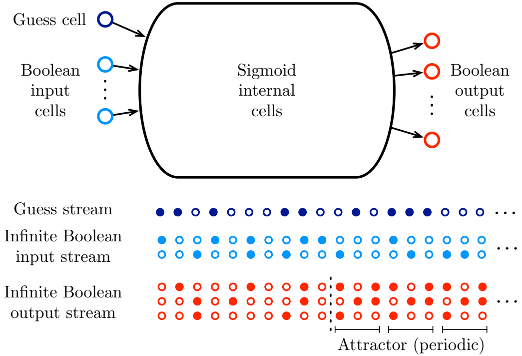
The RNNs are provided with an additional Boolean guess cell.



- Input stream $s \in (\mathbb{B}^M)^\omega$ *accepted* by $\mathcal{N}$ iff there exists some guess $g \in \mathbb{B}^\omega$ s.t. $\mathcal{N}(s, g)$ enters a meaningful attractor.
- Input stream $s \in (\mathbb{B}^M)^\omega$ *rejected* by $\mathcal{N}$ otherwise.

Nondeterminism of Type I

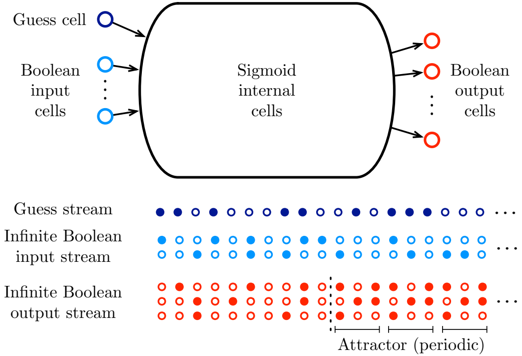
The RNNs are provided with an additional Boolean guess cell.



- Input stream $s \in (\mathbb{B}^M)^\omega$ *accepted* by $\mathcal{N}$ iff there exists some guess $g \in \mathbb{B}^\omega$ s.t. $\mathcal{N}(s, g)$ enters a meaningful attractor.
- Input stream $s \in (\mathbb{B}^M)^\omega$ *rejected* by $\mathcal{N}$ otherwise.

Nondeterminism of Type I

The RNNs are provided with an additional Boolean guess cell.



- Input stream $s \in (\mathbb{B}^M)^\omega$ *accepted* by $\mathcal{N}$ iff there exists some guess $g \in \mathbb{B}^\omega$ s.t. $\mathcal{N}(s, g)$ enters a meaningful attractor.
- Input stream $s \in (\mathbb{B}^M)^\omega$ *rejected* by $\mathcal{N}$ otherwise.

Nondeterminism of Type I

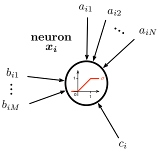
We consider six models of nondeterministic RNNs of type I:

1. static rational RNNs: N-St-RNN[$\mathbb{Q}$]s
2. static real RNNs: N-St-RNN[$\mathbb{R}$]s
3. bi-valued evolving rational RNNs: [illegible]
4. bi-valued evolving real RNNs: [illegible]
5. general evolving rational RNNs: [illegible]
6. general evolving real N-RNNs: [illegible]

Nondeterminism of Type I

We consider six models of nondeterministic RNNs of type I:

1. static rational RNNs: N-St-RNN[$\mathbb{Q}$]s
2. static real RNNs: N-St-RNN[$\mathbb{R}$]s
3. bi-valued evolving rational RNNs: N-Ev$_2$-RNN[$\mathbb{Q}$]s
4. bi-valued evolving real RNNs: N-Ev$_2$-RNN[$\mathbb{R}$]s
5. general evolving rational RNNs: N-Ev-RNN[$\mathbb{Q}$]s
6. general evolving real N-RNNs: N-Ev-RNN[$\mathbb{R}$]s

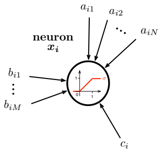


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}\cdot x_j(t) + \sum_{j=1}^{M} b_{ij}\cdot u_j(t) + c_i\right)$$

Nondeterminism of Type I

We consider six models of nondeterministic RNNs of type I:

1. static rational RNNs: N-St-RNN[$\mathbb{Q}$]s
2. static real RNNs: N-St-RNN[$\mathbb{R}$]s
3. bi-valued evolving rational RNNs: N-Ev$_2$-RNN[$\mathbb{Q}$]s
4. bi-valued evolving real RNNs: N-Ev$_2$-RNN[$\mathbb{R}$]s
5. general evolving rational RNNs: N-Ev-RNN[$\mathbb{Q}$]s
6. general evolving real N-RNNs: N-Ev-RNN[$\mathbb{R}$]s

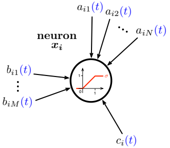


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij} \cdot x_j(t) + \sum_{j=1}^{M} b_{ij} \cdot u_j(t) + c_i\right)$$

Nondeterminism of Type I

We consider six models of nondeterministic RNNs of type I:

1. static rational RNNs: N-St-RNN[$\mathbb{Q}$]s
2. static real RNNs: N-St-RNN[$\mathbb{R}$]s
3. bi-valued evolving rational RNNs: N-Ev$_2$-RNN[$\mathbb{Q}$]s
4. bi-valued evolving real RNNs: N-Ev$_2$-RNN[$\mathbb{R}$]s
5. general evolving rational RNNs: N-Ev-RNN[$\mathbb{Q}$]s
6. general evolving real N-RNNs: N-Ev-RNN[$\mathbb{R}$]s

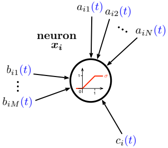


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t)\cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t)\cdot u_j(t) + c_i(t)\right)$$

Nondeterminism of Type I

We consider six models of nondeterministic RNNs of type I:

1. static rational RNNs: N-St-RNN[$\mathbb{Q}$]s
2. static real RNNs: N-St-RNN[$\mathbb{R}$]s
3. bi-valued evolving rational RNNs: N-Ev$_2$-RNN[$\mathbb{Q}$]s
4. bi-valued evolving real RNNs: N-Ev$_2$-RNN[$\mathbb{R}$]s
5. general evolving rational RNNs: N-Ev-RNN[$\mathbb{Q}$]s
6. general evolving real N-RNNs: N-Ev-RNN[$\mathbb{R}$]s

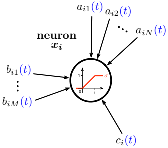


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t)\cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t)\cdot u_j(t) + c_i(t)\right)$$

Nondeterminism of Type I

We consider six models of nondeterministic RNNs of type I:

1. static rational RNNs: N-St-RNN[$\mathbb{Q}$]s
2. static real RNNs: N-St-RNN[$\mathbb{R}$]s
3. bi-valued evolving rational RNNs: N-Ev$_2$-RNN[$\mathbb{Q}$]s
4. bi-valued evolving real RNNs: N-Ev$_2$-RNN[$\mathbb{R}$]s
5. general evolving rational RNNs: N-Ev-RNN[$\mathbb{Q}$]s
6. general evolving real N-RNNs: N-Ev-RNN[$\mathbb{R}$]s

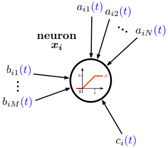


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t)\cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t)\cdot u_j(t) + c_i(t)\right)$$

Nondeterminism of Type I

We consider six models of nondeterministic RNNs of type I:

1. static rational RNNs: N-St-RNN[$\mathbb{Q}$]s
2. static real RNNs: N-St-RNN[$\mathbb{R}$]s
3. bi-valued evolving rational RNNs: N-Ev$_2$-RNN[$\mathbb{Q}$]s
4. bi-valued evolving real RNNs: N-Ev$_2$-RNN[$\mathbb{R}$]s
5. general evolving rational RNNs: N-Ev-RNN[$\mathbb{Q}$]s
6. general evolving real N-RNNs: N-Ev-RNN[$\mathbb{R}$]s



$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t)\cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t)\cdot u_j(t) + c_i(t)\right)$$

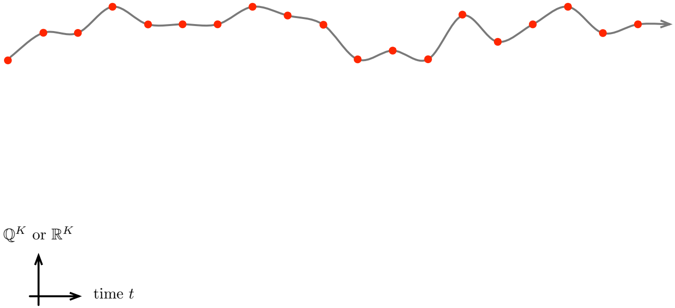
Nondeterminism of Type I

One has the following relationships between those models:

$$
\begin{tikzcd}
 & \text{N-Ev-RNN}[\mathbb{R}]\text{s} \\
\text{N-Ev-RNN}[\mathbb{Q}]\text{s} \arrow[ur] & \\
 & \text{N-Ev}_2\text{-RNN}[\mathbb{R}]\text{s} \arrow[uu] \\
\text{N-Ev}_2\text{-RNN}[\mathbb{Q}]\text{s} \arrow[uu] \arrow[ur] & \\
 & \text{N-St-RNN}[\mathbb{R}]\text{s} \arrow[uu] \\
\text{N-St-RNN}[\mathbb{Q}]\text{s} \arrow[uu] \arrow[ur] &
\end{tikzcd}
$$

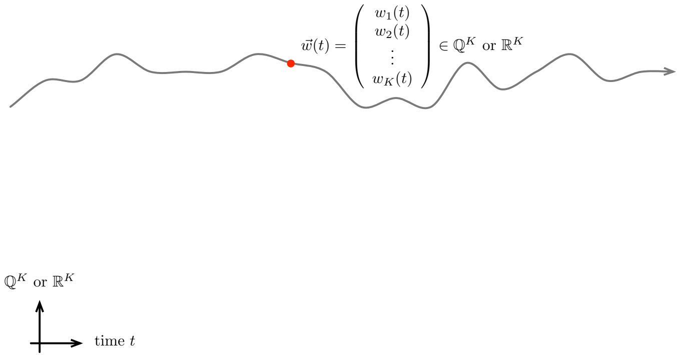
Nondeterminism of Type II

A possible **evolution** $e = \vec{w}(0)\,\vec{w}(1)\,\vec{w}(2)\cdots \in \mathbb{Q}^K$ or $\mathbb{R}^K$



Nondeterminism of Type II

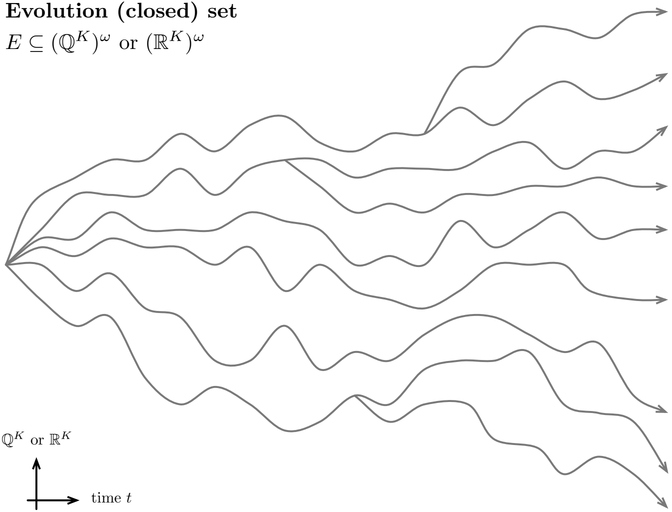
A possible **evolution** $e = \vec{w}(0)\,\vec{w}(1)\,\vec{w}(2) \cdots \in \mathbb{Q}^K$ or $\mathbb{R}^K$



Nondeterminism of Type II

Evolution (closed) set

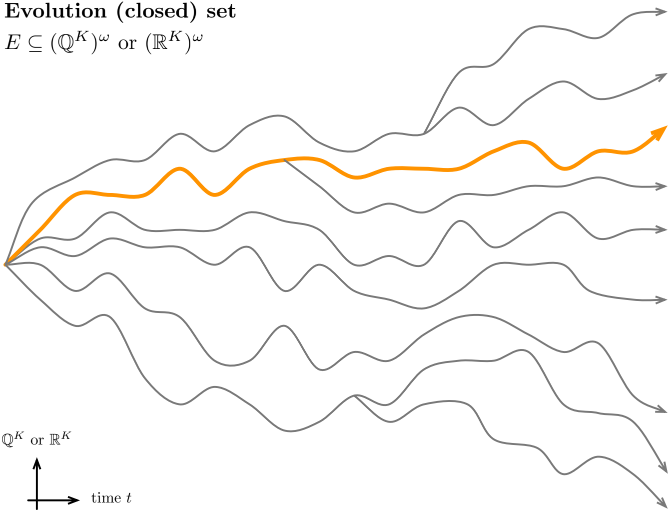
$E \subseteq (\mathbb{Q}^K)^\omega$ or $(\mathbb{R}^K)^\omega$



Nondeterminism of Type II

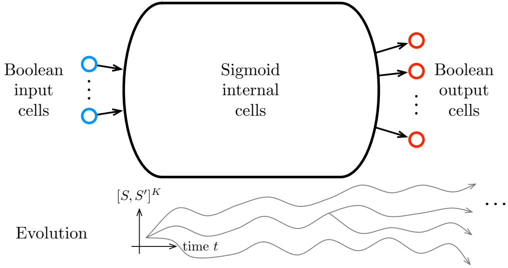
Evolution (closed) set

$E \subseteq (\mathbb{Q}^K)^\omega$ or $(\mathbb{R}^K)^\omega$



Nondeterminism of Type II

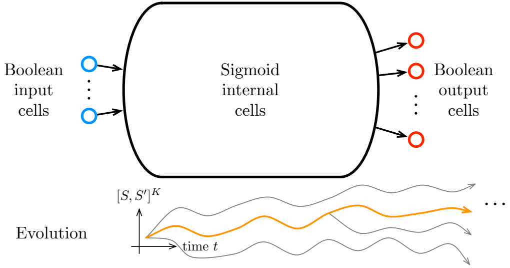
The RNNs are provided with an additional evolution set.



- Input stream $s \in (\mathbb{B}^M)^\omega$ *accepted* by $\mathcal{N}$ iff there exists some evolution $e \in E$ s.t. $\mathcal{N}(s, e)$ enters a meaningful attractor.
- Input stream $s \in (\mathbb{B}^M)^\omega$ is rejected by $\mathcal{N}$

Nondeterminism of Type II

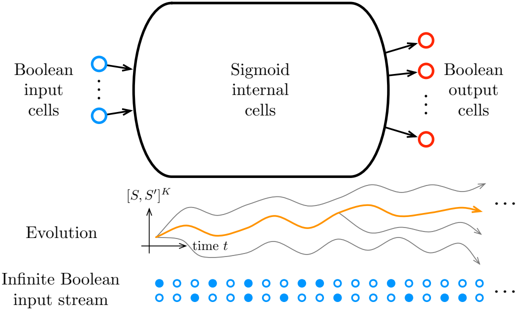
The RNNs are provided with an additional evolution set.



- Input stream $s \in (\mathbb{B}^M)^\omega$ *accepted* by $\mathcal{N}$ iff there exists some evolution $e \in E$ s.t. $\mathcal{N}(s, e)$ enters a meaningful attractor.
- Input stream $s \in (\mathbb{B}^M)^\omega$ *rejected* by $\mathcal{N}$ otherwise.

Nondeterminism of Type II

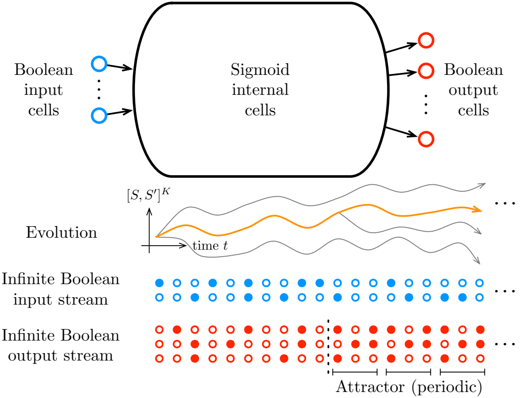
The RNNs are provided with an additional evolution set.



- Input stream $s \in (\mathbb{B}^M)^\omega$ *accepted* by $\mathcal{N}$ iff there exists some evolution $e \in E$ s.t. $\mathcal{N}(s, e)$ enters a meaningful attractor.
- Input stream $s \in (\mathbb{B}^M)^\omega$ *rejected* by $\mathcal{N}$ otherwise.

Nondeterminism of Type II

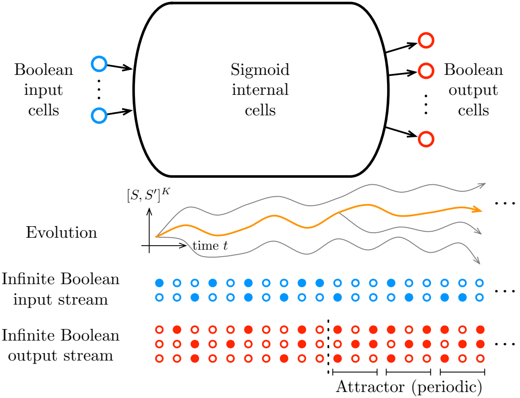
The RNNs are provided with an additional evolution set.



- Input stream $s \in (\mathbb{B}^M)^\omega$ *accepted* by $\mathcal{N}$ iff there exists some evolution $e \in E$ s.t. $\mathcal{N}(s, e)$ enters a meaningful attractor.
- Input stream $s \in (\mathbb{B}^M)^\omega$ *rejected* by $\mathcal{N}$ otherwise.

Nondeterminism of Type II

The RNNs are provided with an additional evolution set.



- Input stream $s \in (\mathbb{B}^M)^\omega$ *accepted* by $\mathcal{N}$ iff there exists some evolution $e \in E$ s.t. $\mathcal{N}(s, e)$ enters a meaningful attractor.
- Input stream $s \in (\mathbb{B}^M)^\omega$ *rejected* by $\mathcal{N}$ otherwise.

Nondeterminism of Type II

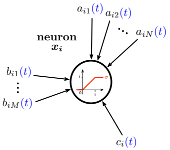
We consider four models of nondeterministic RNNs of type II:

1. bi-valued evolving rational RNNs: $\tilde{N}$-Ev_2-$RNN[\mathbb{Q}]$s
2. bi-valued evolving real RNNs: $\tilde{N}$-Ev_2-$RNN[\mathbb{R}]$s
3. general evolving rational RNNs: $\tilde{N}$-Ev-$RNN[\mathbb{Q}]$s
4. general evolving real N-RNNs: $\tilde{N}$-Ev-$RNN[\mathbb{R}]$s

Nondeterminism of Type II

We consider four models of nondeterministic RNNs of type II:

1. bi-valued evolving rational RNNs: $\tilde{\text{N}}$-Ev_2-$\text{RNN}[\mathbb{Q}]$s
2. bi-valued evolving real RNNs: $\tilde{\text{N}}$-Ev_2-$\text{RNN}[\mathbb{R}]$s
3. general evolving rational RNNs: $\tilde{\text{N}}$-Ev-$\text{RNN}[\mathbb{Q}]$s
4. general evolving real N-RNNs: $\tilde{\text{N}}$-Ev-$\text{RNN}[\mathbb{R}]$s

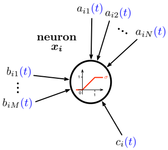


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t) \cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t) \cdot u_j(t) + c_i(t)\right)$$

Nondeterminism of Type II

We consider four models of nondeterministic RNNs of type II:

1. bi-valued evolving rational RNNs: $\tilde{\text{N}}$-Ev$_2$-RNN[$\mathbb{Q}$]s
2. bi-valued evolving real RNNs: $\tilde{\text{N}}$-Ev$_2$-RNN[$\mathbb{R}$]s
3. general evolving rational RNNs: $\tilde{\text{N}}$-Ev-RNN[$\mathbb{Q}$]s
4. general evolving real N-RNNs: $\tilde{\text{N}}$-Ev-RNN[$\mathbb{R}$]s

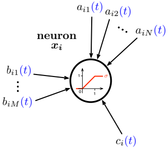


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t) \cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t) \cdot u_j(t) + c_i(t)\right)$$

Nondeterminism of Type II

We consider four models of nondeterministic RNNs of type II:

1. bi-valued evolving rational RNNs: $\tilde{\text{N}}$-Ev$_2$-RNN[$\mathbb{Q}$]s
2. bi-valued evolving real RNNs: $\tilde{\text{N}}$-Ev$_2$-RNN[$\mathbb{R}$]s
3. general evolving rational RNNs: $\tilde{\text{N}}$-Ev-RNN[$\mathbb{Q}$]s
4. general evolving real N-RNNs: $\tilde{\text{N}}$-Ev-RNN[$\mathbb{R}$]s

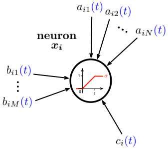


$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t) \cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t) \cdot u_j(t) + c_i(t)\right)$$

Nondeterminism of Type II

We consider four models of nondeterministic RNNs of type II:

1. bi-valued evolving rational RNNs: $\tilde{\text{N}}$-Ev_2-RNN[$\mathbb{Q}$]s
2. bi-valued evolving real RNNs: $\tilde{\text{N}}$-Ev_2-RNN[$\mathbb{R}$]s
3. general evolving rational RNNs: $\tilde{\text{N}}$-Ev-RNN[$\mathbb{Q}$]s
4. general evolving real N-RNNs: $\tilde{\text{N}}$-Ev-RNN[$\mathbb{R}$]s



$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t) \cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t) \cdot u_j(t) + c_i(t)\right)$$

Nondeterminism of Type II

One has the following relationships between those models:

$$
\begin{tikzcd}
\tilde{\mathrm{N}}\text{-Ev-RNN}[\mathbb{Q}]\text{s} \arrow[r, dashed] & \tilde{\mathrm{N}}\text{-Ev-RNN}[\mathbb{R}]\text{s} \\
\tilde{\mathrm{N}}\text{-Ev}_2\text{-RNN}[\mathbb{Q}]\text{s} \arrow[u, dashed] \arrow[r, dashed] & \tilde{\mathrm{N}}\text{-Ev}_2\text{-RNN}[\mathbb{R}]\text{s} \arrow[u, dashed]
\end{tikzcd}
$$

Nondeterminism of Types I and II

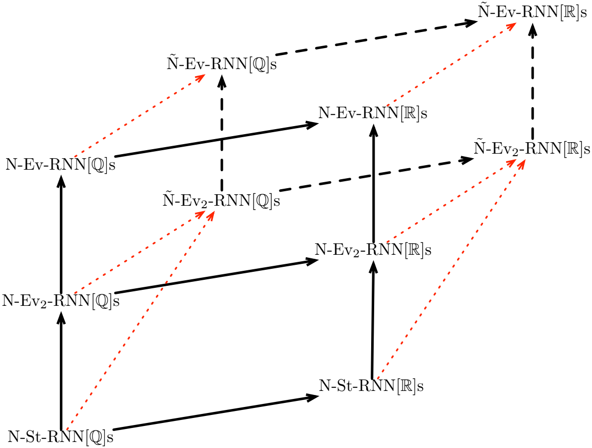
Lemma

The nondeterminism of type I is a particular case of that of type II.

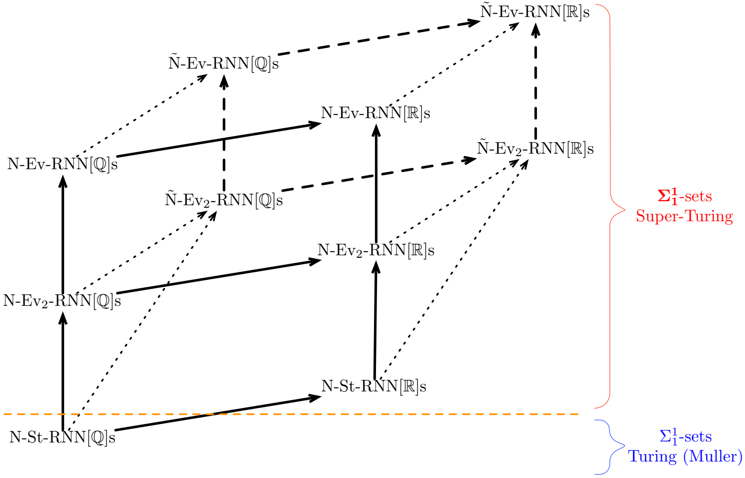
Proof: simply build an evolution set that takes into account all possible guess streams.

Nondeterminism of Types I and II

One has the following relationships between all the models:



Results


$\tilde{\text{N}}$-Ev-RNN[$\mathbb{R}$]s
$\tilde{\text{N}}$-Ev-RNN[$\mathbb{Q}$]s
N-Ev-RNN[$\mathbb{R}$]s
$\tilde{\text{N}}$-Ev$_2$-RNN[$\mathbb{R}$]s
N-Ev-RNN[$\mathbb{Q}$]s
$\tilde{\text{N}}$-Ev$_2$-RNN[$\mathbb{Q}$]s
N-Ev$_2$-RNN[$\mathbb{R}$]s
N-Ev$_2$-RNN[$\mathbb{Q}$]s
N-St-RNN[$\mathbb{R}$]s
N-St-RNN[$\mathbb{Q}$]s
$\boldsymbol{\Sigma^1_1}$-sets
Super-Turing
Σ^1_1-sets
Turing (Muller)

Results

Theorem

Let $L \subseteq (\mathbb{B}^M)^\omega$. The following conditions are equivalent.

- $L \in \Sigma^1_1$
- L *is recognizable by some nondeterministic Muller TM*
- L *is recognizable by some N-St-RNN[$\mathbb{Q}$]*

Proof: Generalization of the classical equivalence between TMs and St-RNN[$\mathbb{Q}$] (Siegelmann & Sontag 95).

Results

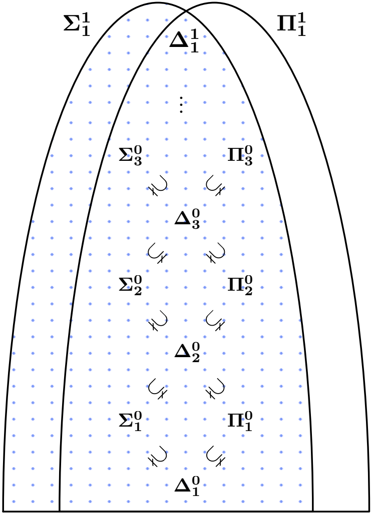
Theorem

Let $L \subseteq (\mathbb{B}^M)^\omega$. The following conditions are equivalent.

- $L \in \Sigma^1_1$
- L *is recognizable by some nondeterministic Muller TM*
- L *is recognizable by some N-St-RNN[$\mathbb{Q}$]*

Proof: Generalization of the classical equivalence between TMs and St-RNN[$\mathbb{Q}$] (Siegelmann & Sontag 95).

Results


$\mathbf{\Sigma}^1_1$
$\mathbf{\Pi}^1_1$
$\mathbf{\Delta}^1_1$
$\vdots$
$\mathbf{\Sigma}^0_3$
$\mathbf{\Pi}^0_3$
$\mathbf{\Delta}^0_3$
$\mathbf{\Sigma}^0_2$
$\mathbf{\Pi}^0_2$
$\mathbf{\Delta}^0_2$
$\mathbf{\Sigma}^0_1$
$\mathbf{\Pi}^0_1$
$\mathbf{\Delta}^0_1$

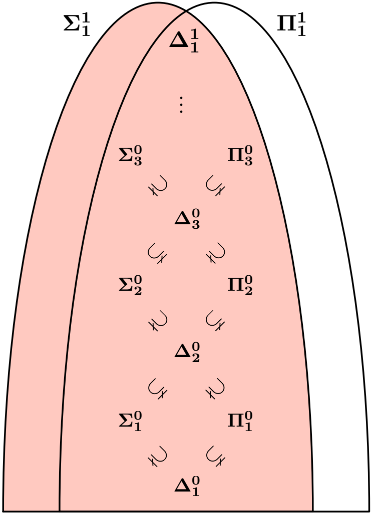
Results

Theorem

Let $L \subseteq (\mathbb{B}^M)^\omega$. The following conditions are equivalent.

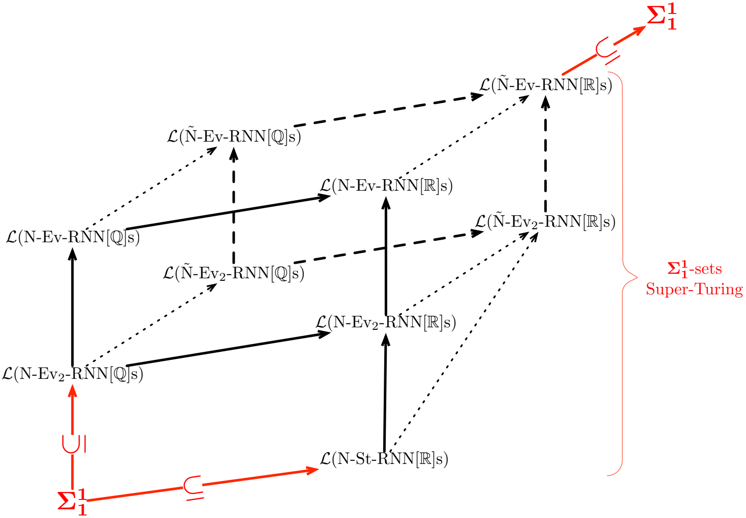
- $L \in \mathbf{\Sigma}^1_1$;
- L *is recognizable by some N-St-RNN[$\mathbb{R}$];*
- L *is recognizable by some N-Ev$_2$-RNN[$\mathbb{Q}$];*
- L *is recognizable by some Ñ-Ev$_2$-RNN[$\mathbb{Q}$];*
- L *is recognizable by some N-Ev-RNN[$\mathbb{Q}$];*
- L *is recognizable by some Ñ-Ev-RNN[$\mathbb{Q}$];*
- L *is recognizable by some N-Ev$_2$-RNN[$\mathbb{R}$];*
- L *is recognizable by some Ñ-Ev$_2$-RNN[$\mathbb{R}$];*
- L *is recognizable by some N-Ev-RNN[$\mathbb{R}$].*
- L *is recognizable by some Ñ-Ev-RNN[$\mathbb{R}$].*

Results



$\mathbf{\Sigma}^1_1$
$\mathbf{\Pi}^1_1$
$\mathbf{\Delta}^1_1$
$\vdots$
$\mathbf{\Sigma}^0_3$
$\mathbf{\Pi}^0_3$
$\mathbf{\Delta}^0_3$
$\mathbf{\Sigma}^0_2$
$\mathbf{\Pi}^0_2$
$\mathbf{\Delta}^0_2$
$\mathbf{\Sigma}^0_1$
$\mathbf{\Pi}^0_1$
$\mathbf{\Delta}^0_1$

Results


Σ^1_1
$\subsetneq$
$\mathcal{L}$(Ñ-Ev-RNN[$\mathbb{R}$]s)
$\mathcal{L}$(Ñ-Ev-RNN[$\mathbb{Q}$]s)
$\mathcal{L}$(N-Ev-RNN[$\mathbb{R}$]s)
$\mathcal{L}$(Ñ-Ev$_2$-RNN[$\mathbb{R}$]s)
$\mathcal{L}$(N-Ev-RNN[$\mathbb{Q}$]s)
$\mathcal{L}$(Ñ-Ev$_2$-RNN[$\mathbb{Q}$]s)
Σ^1_1-sets
Super-Turing
$\mathcal{L}$(N-Ev$_2$-RNN[$\mathbb{R}$]s)
$\mathcal{L}$(N-Ev$_2$-RNN[$\mathbb{Q}$]s)
$\subsetneq$
$\mathcal{L}$(N-St-RNN[$\mathbb{R}$]s)
Σ^1_1
$\subsetneq$

Results

Proposition

Let $L \in \mathbf{\Sigma}^1_1$. Then L is recognizable by some N-Ev$_2$-RNN[$\mathbb{Q}$] or by some N-St-RNN[$\mathbb{R}$].

Proof (sketch):

- Since $L \in \Sigma^1_1$, there exists some Π^0_2 set $X \subseteq (\mathbb{B}^M)^\omega \times \{0,1\}^\omega$ such that $L = \pi_1(X)$.

$$X = \bigcap_{i \geq 0} \bigcup_{j \geq 0} \left(p_{i,j} \cdot (\mathbb{B}^M)^\omega \times q_{i,j} \cdot \{0,1\}^\omega \right)$$

- X can be recursively encoded into some $r_X \in \mathbb{R}$.

Results

Proposition

Let $L \in \mathbf{\Sigma}^1_1$. Then L is recognizable by some N-Ev$_2$-RNN[$\mathbb{Q}$] or by some N-St-RNN[$\mathbb{R}$].

Proof (sketch):

- Since $L \in \mathbf{\Sigma}^1_1$, there exists some $\mathbf{\Pi}^0_2$ set $X \subseteq (\mathbb{B}^M)^\omega \times \{0,1\}^\omega$ such that $L = \pi_1(X)$.

$$X = \bigcap_{i \geq 0} \bigcup_{j \geq 0} \left(p_{i,j} \cdot (\mathbb{B}^M)^\omega \times q_{i,j} \cdot \{0,1\}^\omega \right)$$

- X can be recursively encoded into some $r_X \in \mathbb{R}$.

Results

Proposition

Let $L \in \mathbf{\Sigma}^1_1$. Then L is recognizable by some N-Ev$_2$-RNN[$\mathbb{Q}$] or by some N-St-RNN[$\mathbb{R}$].

Proof (sketch):

- Since $L \in \mathbf{\Sigma}^1_1$, there exists some $\mathbf{\Pi}^0_2$ set $X \subseteq (\mathbb{B}^M)^\omega \times \{0,1\}^\omega$ such that $L = \pi_1(X)$.

$$X = \bigcap_{i \geq 0} \bigcup_{j \geq 0} \left(p_{i,j} \cdot (\mathbb{B}^M)^\omega \times q_{i,j} \cdot \{0,1\}^\omega \right)$$

- X can be recursively encoded into some $r_X \in \mathbb{R}$.

Results

Algorithm 1 Infinite procedure

Require: Input stream $s = \vec{u}(0)\vec{u}(1)\vec{u}(2)\cdots \in (\mathbb{B}^M)^\omega$,
guess stream $g = g(0)g(1)g(2)\cdots \in \mathbb{B}^\omega$, and real number r_X.

store each $\vec{u}(t) \in \mathbb{B}^M$ and $g(t) \in \{0,1\}$ as they arrive
$i \leftarrow 0$, $j \leftarrow 0$
loop
  decode $(p_{i,j}, q_{i,j})$ from r_X // recursive procedure if r_X is given
  if $p_{i,j} \subseteq s[0{:}c]$ and $q_{i,j} \subseteq g[0{:}c]$ **then**   // $(s,g) \in p_{i,j} \cdot (\mathbb{B}^M)^\omega \times q_{i,j} \cdot \{0,1\}^\omega$
    return 1 // $\exists\ j$ s.t. $(s,g) \in p_{i,j} \cdot (\mathbb{B}^M)^\omega \times q_{i,j} \cdot \{0,1\}^\omega$
    $i \leftarrow i+1$, $j \leftarrow 0$ // test if $(s,g) \in p_{i+1,0} \cdot (\mathbb{B}^M)^\omega \times q_{i+1,0} \cdot \{0,1\}^\omega$
  else   // $(s,g) \notin p_{i,j} \cdot (\mathbb{B}^M)^\omega \times q_{i,j} \cdot \{0,1\}^\omega$
    return 0 // $\neg\exists j' \leq j$ s.t. $(s,g) \in p_{i,j'} \cdot (\mathbb{B}^M)^\omega \times q_{i,j'} \cdot \{0,1\}^\omega$
    $i \leftarrow i$, $j \leftarrow j+1$ // test if $(s,g) \in p_{i,j+1} \cdot (\mathbb{B}^M)^\omega \times q_{i,j+1} \cdot \{0,1\}^\omega$
  end if
end loop

Algo returns ∞-many 1's iff $(s,g) \in \bigcap_i \bigcup_j (p_{i,j} \cdot (\mathbb{B}^M)^\omega \times q_{i,j} \cdot \{0,1\}^\omega) = X$.

Results

Algorithm 2 Infinite procedure

Require: Input stream $s = \vec{u}(0)\vec{u}(1)\vec{u}(2)\cdots \in (\mathbb{B}^M)^\omega$,
guess stream $g = g(0)g(1)g(2)\cdots \in \mathbb{B}^\omega$, and real number r_X.

```
store each u⃗(t) ∈ 𝔹^M and g(t) ∈ {0,1} as they arrive
i ← 0, j ← 0
loop
    decode (p_{i,j}, q_{i,j}) from r_X             // recursive procedure if r_X is given
    if p_{i,j} ⊆ s[0:c] and q_{i,j} ⊆ g[0:c] then  // (s,g) ∈ p_{i,j}·(𝔹^M)^ω × q_{i,j}·{0,1}^ω
        return 1                                   // ∃ j s.t. (s,g) ∈ p_{i,j}·(𝔹^M)^ω × q_{i,j}·{0,1}^ω
        i ← i+1, j ← 0                             // test if (s,g) ∈ p_{i+1,0}·(𝔹^M)^ω × q_{i+1,0}·{0,1}^ω
    else                                           // (s,g) ∉ p_{i,j}·(𝔹^M)^ω × q_{i,j}·{0,1}^ω
        return 0                                   // ¬∃j' ≤ j s.t. (s,g) ∈ p_{i,j'}·(𝔹^M)^ω × q_{i,j'}·{0,1}^ω
        i ← i, j ← j+1                             // test if (s,g) ∈ p_{i,j+1}·(𝔹^M)^ω × q_{i,j+1}·{0,1}^ω
    end if
end loop
```

Algo returns ∞-many 1's iff $(s,g) \in \bigcap_i \bigcup_j (p_{i,j} \cdot (\mathbb{B}^M)^\omega \times q_{i,j} \cdot \{0,1\}^\omega) = X$.

Results

- The algo can be simulated by some N-St-RNN[$\mathbb{R}$] $\mathcal{N}$, i.e.,

  algo returns infinitely many 1's on (s, g)
  iff
  $\mathcal{N}(s, g)$ visits a meaningful attractor.

- $s \in L(\mathcal{N})$
  - iff $\exists\, g \in \mathbb{B}^\omega$ s.t. $\mathcal{N}(s, g)$ visits a meaningful attractor
  - iff $\exists\, g \in \mathbb{B}^\omega$ s.t. the algo returns infinitely many 1's on (s, g)
  - iff $\exists\, g \in \mathbb{B}^\omega$ s.t. $(s, g) \in X$
  - iff $s \in \pi_1(X) = L$.

- Therefore, $L(\mathcal{N}) = L$. $\square$

Results

- The algo can be simulated by some N-St-RNN[$\mathbb{R}$] $\mathcal{N}$, i.e.,

  algo returns infinitely many 1's on (s, g)
  iff
  $\mathcal{N}(s, g)$ visits a meaningful attractor.

- $s \in L(\mathcal{N})$
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. $\mathcal{N}(s, g)$ visits a meaningful attractor
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. the algo returns infinitely many 1's on (s, g)
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. $(s, g) \in X$
  - iff $s \in \pi_1(X) = L$.

- Therefore, $L(\mathcal{N}) = L$. □

Results

- The algo can be simulated by some N-St-RNN[$\mathbb{R}$] $\mathcal{N}$, i.e.,

  algo returns infinitely many 1's on (s, g)

  iff

  $\mathcal{N}(s, g)$ visits a meaningful attractor.

- $s \in L(\mathcal{N})$
  - iff $\exists\, g \in \mathbb{B}^\omega$ s.t. $\mathcal{N}(s, g)$ visits a meaningful attractor
  - iff $\exists\, g \in \mathbb{B}^\omega$ s.t. the algo returns infinitely many 1's on (s, g)
  - iff $\exists\, g \in \mathbb{B}^\omega$ s.t. $(s, g) \in X$
  - iff $s \in \pi_1(X) = L$.

- Therefore, $L(\mathcal{N}) = L$. $\square$

Results

- The algo can be simulated by some N-St-RNN[$\mathbb{R}$] $\mathcal{N}$, i.e.,

  algo returns infinitely many 1's on (s, g)

  iff

  $\mathcal{N}(s, g)$ visits a meaningful attractor.

- $s \in L(\mathcal{N})$
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. $\mathcal{N}(s, g)$ visits a meaningful attractor
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. the algo returns infinitely many 1's on (s, g)
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. $(s, g) \in X$
  - iff $s \in \pi_1(X) = L$.

- Therefore, $L(\mathcal{N}) = L$. $\square$

Results

- The algo can be simulated by some N-St-RNN[$\mathbb{R}$] $\mathcal{N}$, i.e.,

  algo returns infinitely many 1's on (s, g)

  iff

  $\mathcal{N}(s, g)$ visits a meaningful attractor.

- $s \in L(\mathcal{N})$
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. $\mathcal{N}(s, g)$ visits a meaningful attractor
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. the algo returns infinitely many 1's on (s, g)
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. $(s, g) \in X$
  - iff $s \in \pi_1(X) = L$.
- Therefore, $L(\mathcal{N}) = L$. $\square$

Results

- The algo can be simulated by some N-St-RNN[$\mathbb{R}$] $\mathcal{N}$, i.e.,

$$\text{algo returns infinitely many } 1\text{'s on } (s,g)$$
$$\text{iff}$$
$$\mathcal{N}(s,g) \text{ visits a meaningful attractor.}$$

- $s \in L(\mathcal{N})$
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. $\mathcal{N}(s,g)$ visits a meaningful attractor
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. the algo returns infinitely many 1's on (s,g)
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. $(s,g) \in X$
  - iff $s \in \pi_1(X) = L$.
- Therefore, $L(\mathcal{N}) = L$. $\square$

Results

- The algo can be simulated by some N-St-RNN[$\mathbb{R}$] $\mathcal{N}$, i.e.,

  algo returns infinitely many 1's on (s, g)
  iff
  $\mathcal{N}(s, g)$ visits a meaningful attractor.

- $s \in L(\mathcal{N})$
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. $\mathcal{N}(s, g)$ visits a meaningful attractor
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. the algo returns infinitely many 1's on (s, g)
  - iff $\exists\ g \in \mathbb{B}^\omega$ s.t. $(s, g) \in X$
  - iff $s \in \pi_1(X) = L$.
- Therefore, $L(\mathcal{N}) = L$. $\square$

Results

Proposition

Let $\mathcal{N}$ be some $\tilde{\mathrm{N}}$-Ev-RNN[$\mathbb{R}$]. Then $L(\mathcal{N}) \in \mathbf{\Sigma}^1_1$.

Proof (sketch):

- The function $f_{\mathcal{N}} : (\mathbb{B}^M)^\omega \times E \to (\mathbb{B}^P)^\omega$ associated with the dynamics of $\mathcal{N}$ is of Baire class 1 (preimage of a $\mathbf{\Sigma}^0_1$ is a $\mathbf{\Sigma}^0_2$).
- Accordingly, the ω-language $L(\mathcal{N})$ can be expressed as the first projection of a finite Boolean combination of $\mathbf{\Sigma}^0_3$ and $\mathbf{\Pi}^0_3$ sets (i.e., of a Borel set) of the Polish space $(\mathbb{B}^M)^\omega \times E$.
- Therefore, $L(\mathcal{N}) \in \mathbf{\Sigma}^1_1$. □

Results

Proposition

Let $\mathcal{N}$ be some Ñ-Ev-RNN[$\mathbb{R}$]. Then $L(\mathcal{N}) \in \mathbf{\Sigma}^1_1$.

Proof (sketch):

- The function $f_{\mathcal{N}} : (\mathbb{B}^M)^\omega \times E \to (\mathbb{B}^P)^\omega$ associated with the dynamics of $\mathcal{N}$ is of Baire class 1 (preimage of a $\mathbf{\Sigma}^0_1$ is a $\mathbf{\Sigma}^0_2$).
- Accordingly, the ω-language $L(\mathcal{N})$ can be expressed as the first projection of a finite Boolean combination of $\mathbf{\Sigma}^0_3$ and $\mathbf{\Pi}^0_3$ sets (i.e., of a Borel set) of the Polish space $(\mathbb{B}^M)^\omega \times E$.
- Therefore, $L(\mathcal{N}) \in \mathbf{\Sigma}^1_1$. $\square$

Results

Proposition

Let $\mathcal{N}$ be some Ñ-Ev-RNN[$\mathbb{R}$]. Then $L(\mathcal{N}) \in \mathbf{\Sigma^1_1}$.

Proof (sketch):

- The function $f_{\mathcal{N}} : (\mathbb{B}^M)^\omega \times E \to (\mathbb{B}^P)^\omega$ associated with the dynamics of $\mathcal{N}$ is of Baire class 1 (preimage of a $\mathbf{\Sigma}^0_1$ is a $\mathbf{\Sigma}^0_2$).
- Accordingly, the ω-language $L(\mathcal{N})$ can be expressed as the first projection of a finite Boolean combination of $\mathbf{\Sigma^0_3}$ and $\mathbf{\Pi^0_3}$ sets (i.e., of a Borel set) of the Polish space $(\mathbb{B}^M)^\omega \times E$.
- Therefore, $L(\mathcal{N}) \in \mathbf{\Sigma}^1_1$. $\square$

Results

Proposition

Let $\mathcal{N}$ be some Ñ-Ev-RNN[$\mathbb{R}$]. Then $L(\mathcal{N}) \in \mathbf{\Sigma}_1^1$.

Proof (sketch):

- The function $f_{\mathcal{N}} : (\mathbb{B}^M)^\omega \times E \to (\mathbb{B}^P)^\omega$ associated with the dynamics of $\mathcal{N}$ is of Baire class 1 (preimage of a $\mathbf{\Sigma}_1^0$ is a $\mathbf{\Sigma}_2^0$).
- Accordingly, the ω-language $L(\mathcal{N})$ can be expressed as the first projection of a finite Boolean combination of $\mathbf{\Sigma}_3^0$ and $\mathbf{\Pi}_3^0$ sets (i.e., of a Borel set) of the Polish space $(\mathbb{B}^M)^\omega \times E$.
- Therefore, $L(\mathcal{N}) \in \mathbf{\Sigma}_1^1$. $\square$

Results – Summary

NONDET.	Static	Bi-valued Evolving	General Evolving
$\mathbb{Q}$	N-St-RNN[$\mathbb{Q}$]s	N-Ev$_2$-RNN[$\mathbb{Q}$]s	N-Ev-RNN[$\mathbb{Q}$]s
	–	Ñ-Ev$_2$-RNN[$\mathbb{Q}$]s	Ñ-Ev-RNN[$\mathbb{Q}$]s
	$= \Sigma^1_1$ (lightface)	$= \boldsymbol{\Sigma}^1_1$ (boldface)	$= \boldsymbol{\Sigma}^1_1$ (boldface)
	Turing (Muller)	super-Turing	super-Turing
$\mathbb{R}$	N-St-RNN[$\mathbb{R}$]s	N-Ev$_2$-RNN[$\mathbb{R}$]s	N-Ev-RNN[$\mathbb{R}$]s
	–	Ñ-Ev$_2$-RNN[$\mathbb{R}$]s	Ñ-Ev-RNN[$\mathbb{R}$]s
	$= \boldsymbol{\Sigma}^1_1$ (boldface)	$= \boldsymbol{\Sigma}^1_1$ (boldface)	$= \boldsymbol{\Sigma}^1_1$ (boldface)
	super-Turing	super-Turing	super-Turing

Conclusion

- We provided a characterization of the expressive power of recurrent neural networks in terms of their attractor dynamics.
- In general, the super-Turing computational capabilities of neural models raises the question of *hypercomputation*.
- Current physical theories are consistent with the possibility of hypercomputational systems (quantum, relativistic, etc.). No such systems are currently feasible or harnessable.
- Philosophical and scientific literature about hypercomputation is however flourishing.

Conclusion

- We provided a characterization of the expressive power of recurrent neural networks in terms of their attractor dynamics.
- In general, the super-Turing computational capabilities of neural models raises the question of *hypercomputation*.
- Current physical theories are consistent with the possibility of hypercomputational systems (quantum, relativistic, etc.). No such systems are currently feasible or harnessable.
- Philosophical and scientific literature about hypercomputation is however flourishing.

Conclusion

- We provided a characterization of the expressive power of recurrent neural networks in terms of their attractor dynamics.
- In general, the super-Turing computational capabilities of neural models raises the question of *hypercomputation*.
- Current physical theories are consistent with the possibility of hypercomputational systems (quantum, relativistic, etc.). No such systems are currently feasible or harnessable.
- Philosophical and scientific literature about hypercomputation is however flourishing.

Conclusion

- We provided a characterization of the expressive power of recurrent neural networks in terms of their attractor dynamics.
- In general, the super-Turing computational capabilities of neural models raises the question of *hypercomputation*.
- Current physical theories are consistent with the possibility of hypercomputational systems (quantum, relativistic, etc.). No such systems are currently feasible or harnessable.
- Philosophical and scientific literature about hypercomputation is however flourishing.