Computational Capabilities of Recurrent Neural Networks

Jérémie Cabessa

University of Lausanne, Switzerland

5 September 2012

Introduction

- We follow the so-called *mind-computer analogy* approach to cognitive science.
- We study the computational capabilities of basic models of recurrent neural networks.
- We show that recurrent neural networks provide a natural model of computation beyond the Turing limits.

Introduction

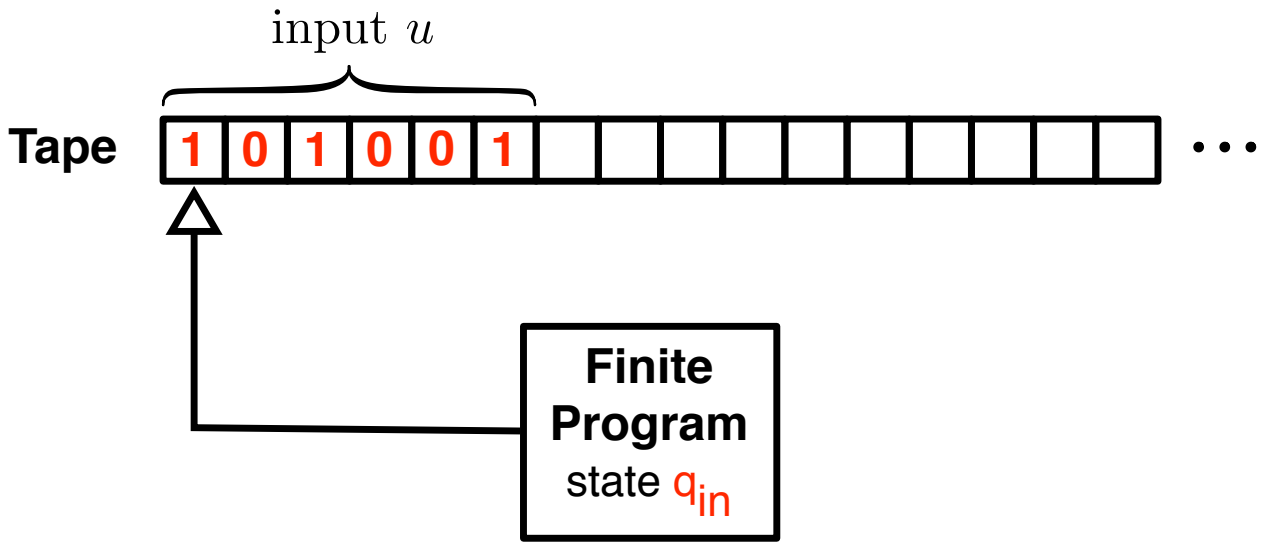
- We follow the so-called *mind-computer analogy* approach to cognitive science.
- We study the computational capabilities of basic models of recurrent neural networks.
- We show that recurrent neural networks provide a natural model of computation beyond the Turing limits.

Introduction

- We follow the so-called *mind-computer analogy* approach to cognitive science.
- We study the computational capabilities of basic models of recurrent neural networks.
- We show that recurrent neural networks provide a natural model of computation beyond the Turing limits.

Turing machine

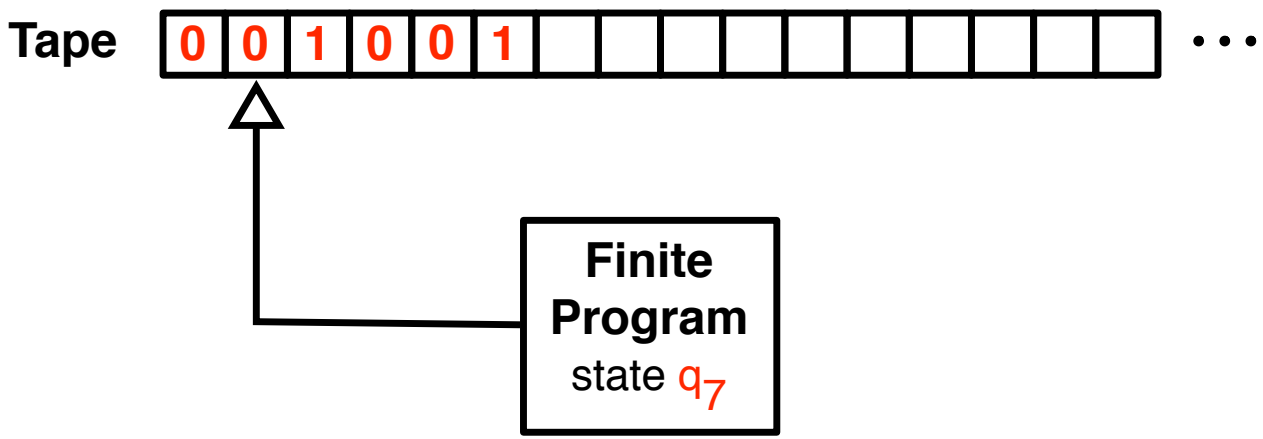
A Turing machine (TM) consists of an infinite tape, a read-write head, and a finite program.



- 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

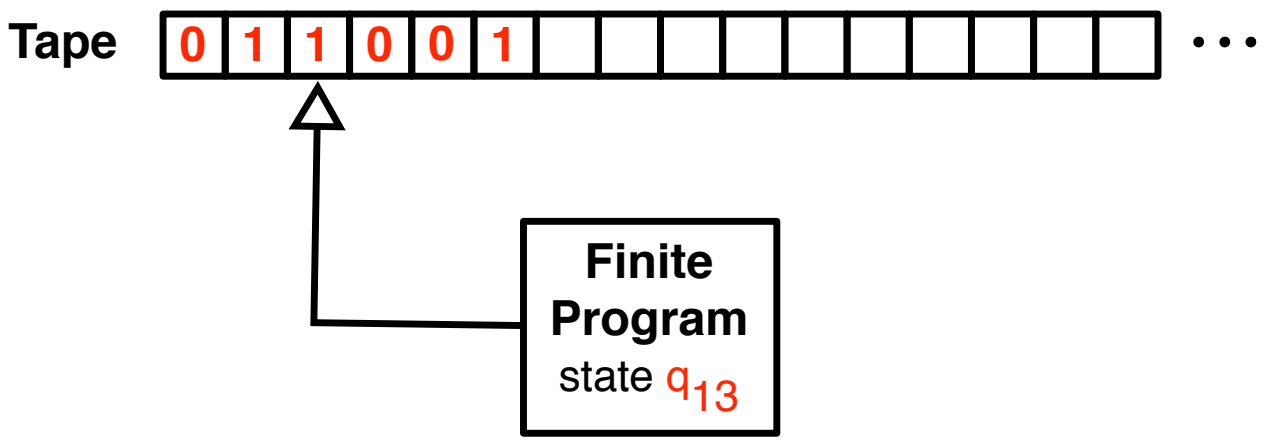
A Turing machine (TM) consists of an infinite tape, a read-write head, and a finite program.



- 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}

Turing machine

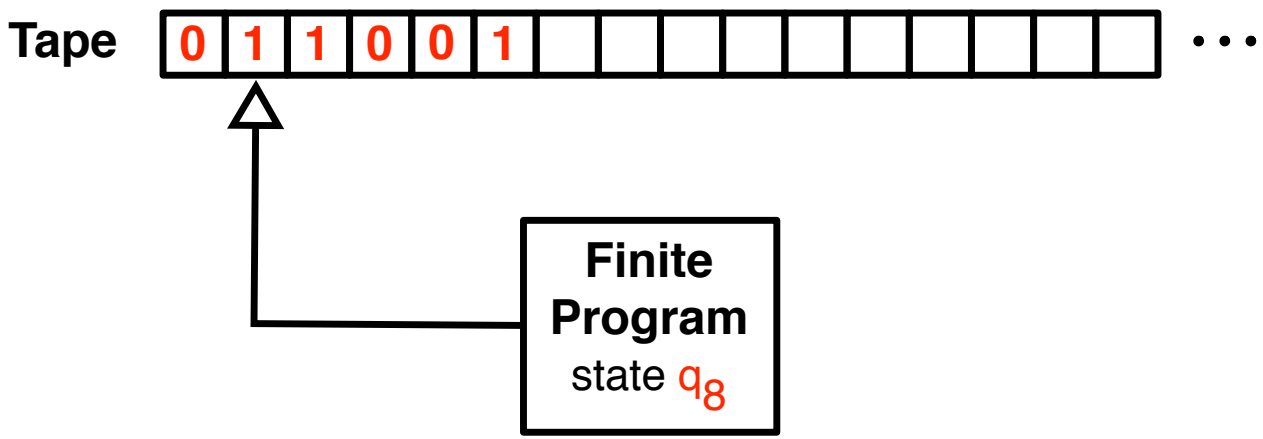
A Turing machine (TM) consists of an infinite tape, a read-write head, and a finite program.



- 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

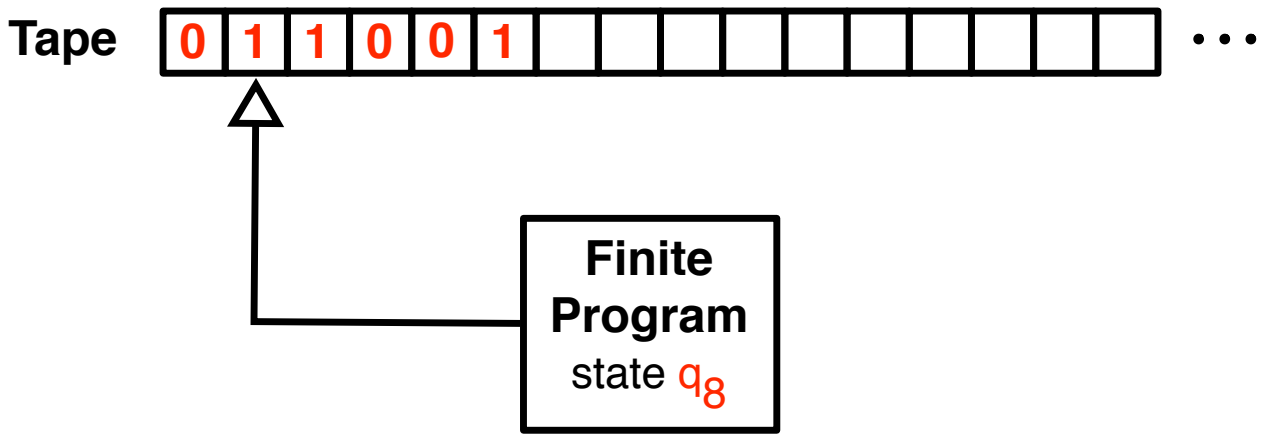
A Turing machine (TM) consists of an infinite tape, a read-write head, and a finite program.



- 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}

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}

The Church-Turing thesis

The Church-Turing Thesis states that the Turing machine model is capable of capturing all possible aspects of algorithmic computation.

> *A function is algorithmically computable if and only if it is computable by a Turing machine.*

Recurrent neural networks actually provide a natural model beyond the Turing limits.

The Church-Turing thesis

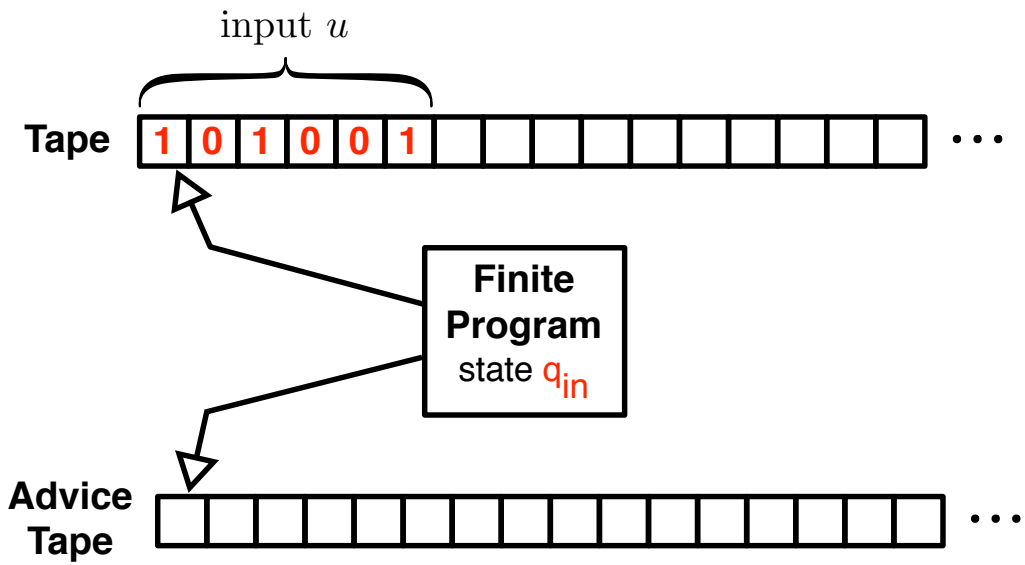
The Church-Turing Thesis states that the Turing machine model is capable of capturing all possible aspects of algorithmic computation.

> *A function is algorithmically computable if and only if it is computable by a Turing machine.*

Recurrent neural networks actually provide a natural model beyond the Turing limits.

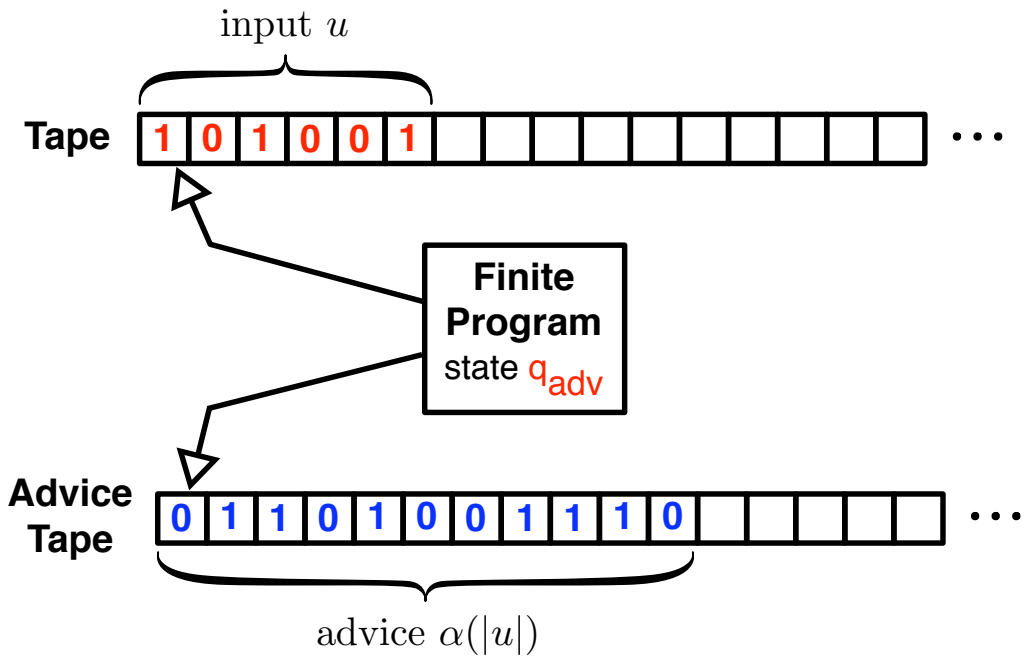
Turing machine with advice

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



Turing machine with advice

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



TM vs TM/A

The complexity class $\mathbf{P}$ ($\mathbf{PTIME}$) is the collections of all languages decidable in polynomial time by some TM.

The complexity class $\mathbf{P/poly}$ is the collections of all languages decidable in polynomial time by some TM/poly(A).

Lemma

TM/poly(A)s are strictly more powerful than TMs (super-Turing) already in polynomial time of computation, i.e. $\mathbf{P/poly} \supsetneq \mathbf{P}$.

TM vs TM/A

The complexity class $\mathbf{P}$ ($\mathbf{PTIME}$) is the collections of all languages decidable in polynomial time by some TM.

The complexity class $\mathbf{P/poly}$ is the collections of all languages decidable in polynomial time by some TM/poly(A).

Lemma

TM/poly(A)s are strictly more powerful than TMs (super-Turing) already in polynomial time of computation, i.e. $\mathbf{P/poly} \supsetneq \mathbf{P}$.

TM vs TM/A

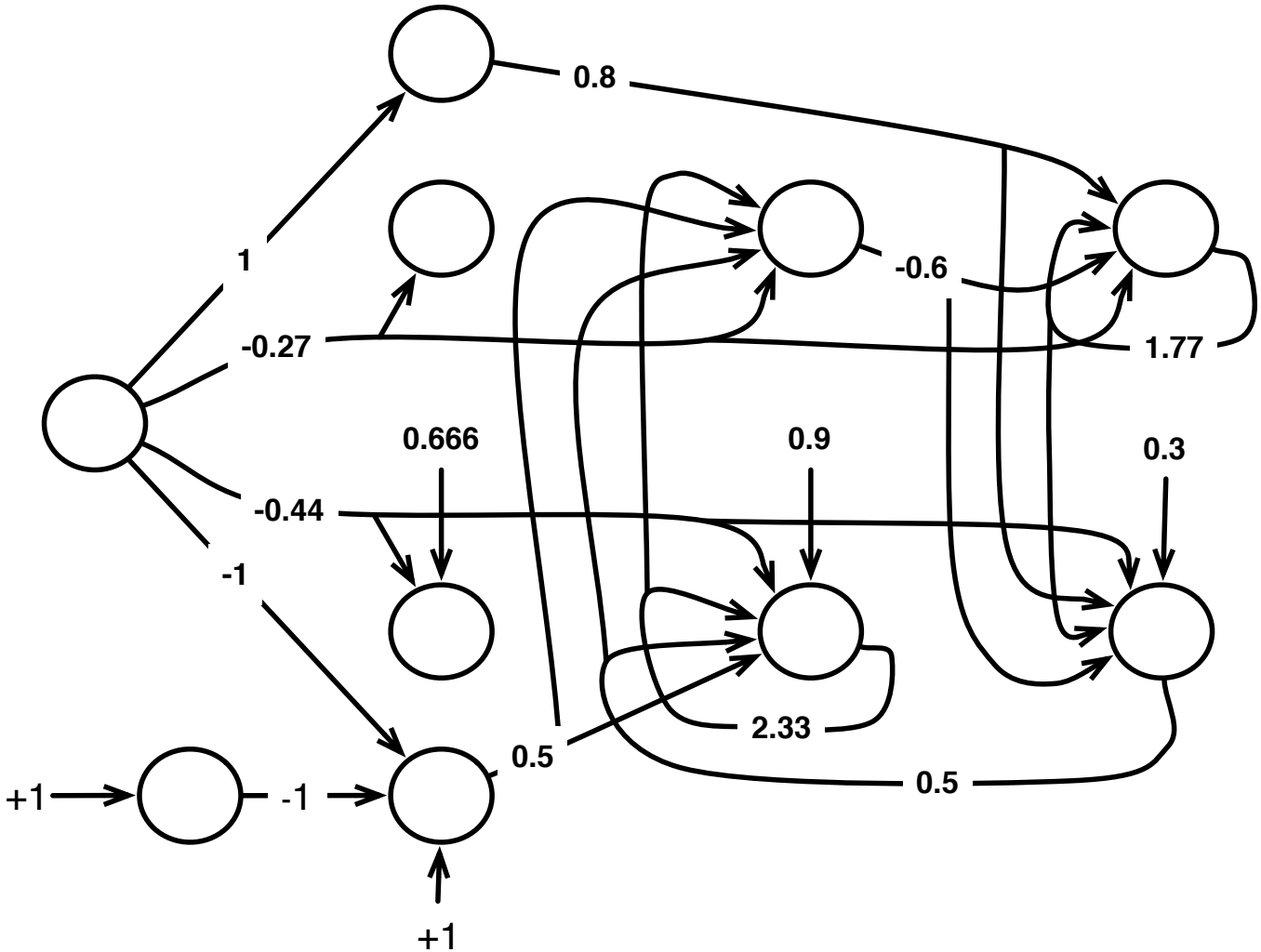
The complexity class $\mathbf{P}$ ($\mathbf{PTIME}$) is the collections of all languages decidable in polynomial time by some TM.

The complexity class $\mathbf{P/poly}$ is the collections of all languages decidable in polynomial time by some TM/poly(A).

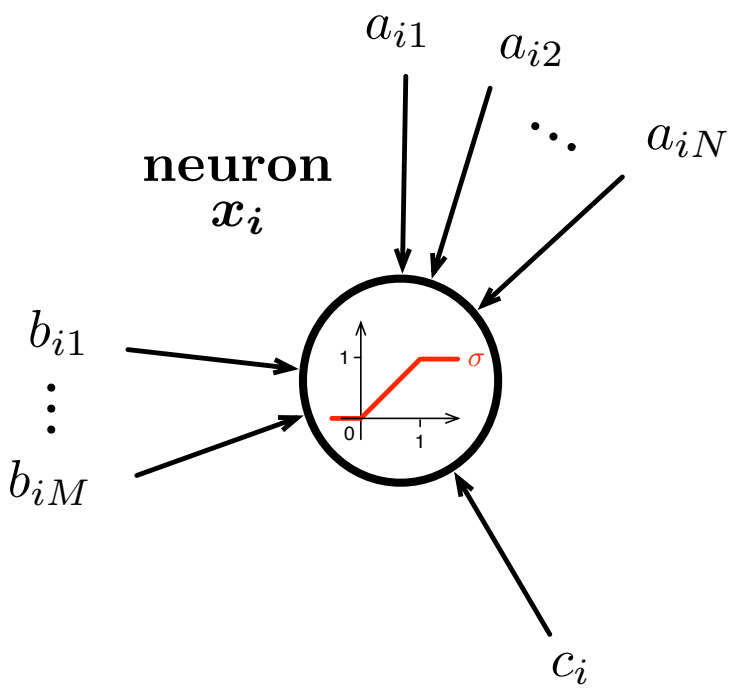
Lemma

TM/poly(A)s are strictly more powerful than TMs (super-Turing) already in polynomial time of computation, i.e. $\mathbf{P/poly} \supsetneq \mathbf{P}$.

Recurrent neural networks



Dynamics: static synaptic weights



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

Computational power

RNN[$\mathbb{Q}$]s: networks with static *rational* weights

RNN[$\mathbb{R}$]s: networks with static *real* (analog) weights

Theorem (Siegelmann & Sontag 94, 95)

- *RNN[$\mathbb{Q}$]s are Turing equivalent.*
- *RNN[$\mathbb{R}$]s are super-Turing, i.e.*
  RNN[$\mathbb{R}$]s are equivalent to TM/poly(A) in poly time.

Computational power

RNN[$\mathbb{Q}$]s: networks with static *rational* weights

RNN[$\mathbb{R}$]s: networks with static *real* (analog) weights

Theorem (Siegelmann & Sontag 94, 95)

- *RNN[$\mathbb{Q}$]s are Turing equivalent.*
- *RNN[$\mathbb{R}$]s are super-Turing, i.e. RNN[$\mathbb{R}$]s are equivalent to TM/poly(A) in poly time.*

Computational power

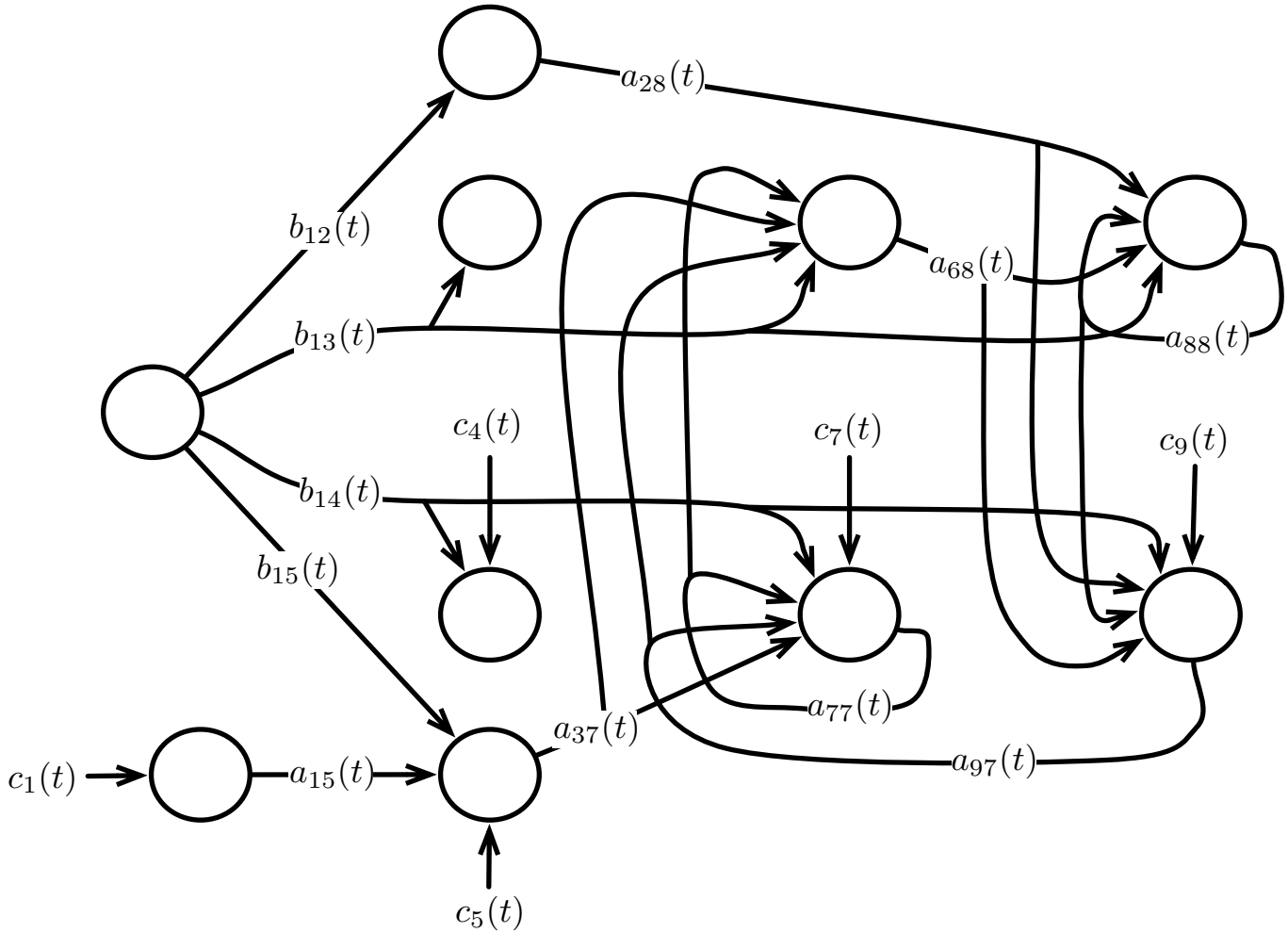
RNN[$\mathbb{Q}$]s: networks with static *rational* weights

RNN[$\mathbb{R}$]s: networks with static *real* (analog) weights

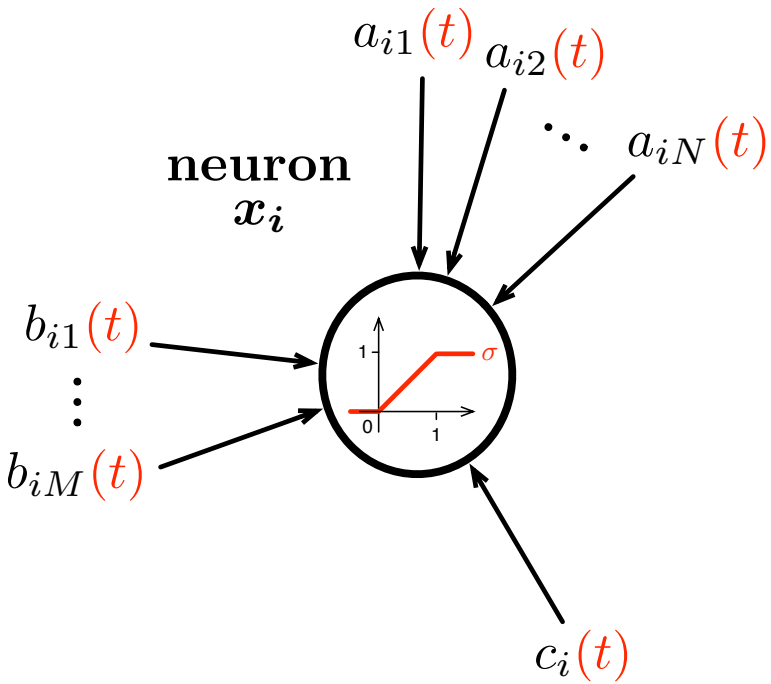
Theorem (Siegelmann & Sontag 94, 95)

- *RNN[$\mathbb{Q}$]s are Turing equivalent.*
- *RNN[$\mathbb{R}$]s are super-Turing, i.e.*
  RNN[$\mathbb{R}$]s are equivalent to TM/poly(A) in poly time.

Evolving recurrent neural networks



Dynamics: evolving synaptic weights



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

Computational power

Ev-RNN[$\mathbb{Q}$]s: networks with evolving *rational* weights

Ev-RNN[$\mathbb{R}$]s: networks with evolving *real* (analog) weights

Theorem (Cabessa & Siegelmann 11)

- *Ev-RNN[$\mathbb{Q}$]s are super-Turing, i.e. Ev-RNN[$\mathbb{Q}$]s are equivalent to TM/poly(A) in poly time.*
- *Ev-RNN[$\mathbb{R}$]s are super-Turing, i.e. Ev-RNN[$\mathbb{R}$]s are equivalent to TM/poly(A) in poly time.*

Computational power

Ev-RNN[$\mathbb{Q}$]s: networks with evolving *rational* weights

Ev-RNN[$\mathbb{R}$]s: networks with evolving *real* (analog) weights

Theorem (Cabessa & Siegelmann 11)

- *Ev-RNN[$\mathbb{Q}$]s are super-Turing, i.e.*
  Ev-RNN[$\mathbb{Q}$]s are equivalent to TM/poly(A) in poly time.
- *Ev-RNN[$\mathbb{R}$]s are super-Turing, i.e.*
  Ev-RNN[$\mathbb{R}$]s are equivalent to TM/poly(A) in poly time.

Computational power

Ev-RNN[$\mathbb{Q}$]s: networks with evolving *rational* weights

Ev-RNN[$\mathbb{R}$]s: networks with evolving *real* (analog) weights

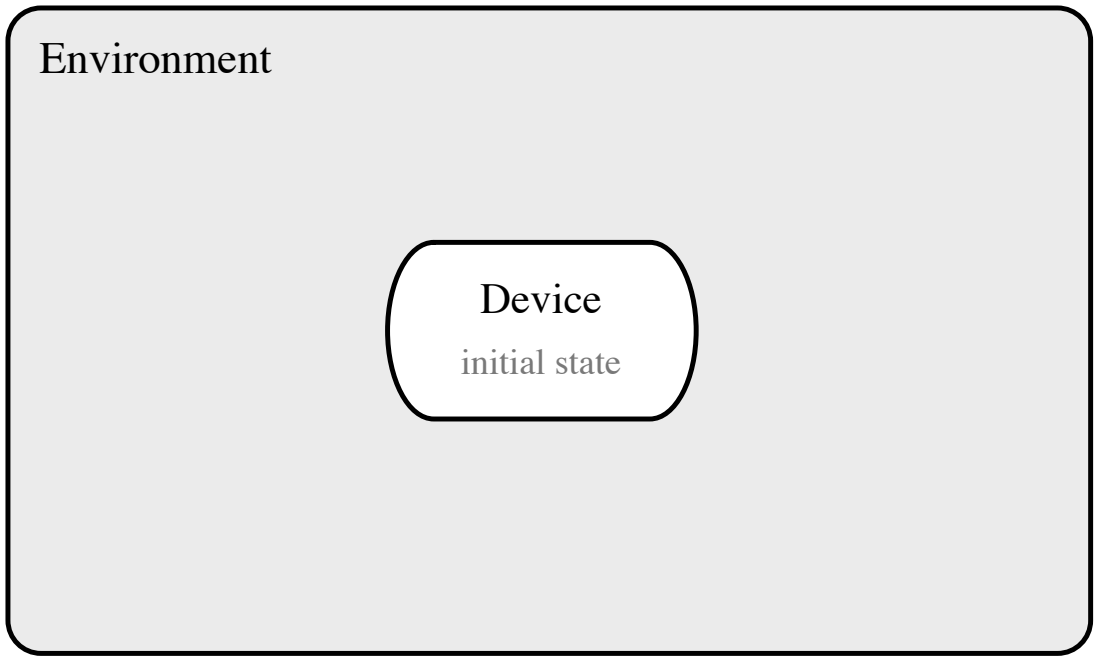
Theorem (Cabessa & Siegelmann 11)

- *Ev-RNN[$\mathbb{Q}$]s are super-Turing, i.e.*
  Ev-RNN[$\mathbb{Q}$]s are equivalent to TM/poly(A) in poly time.
- *Ev-RNN[$\mathbb{R}$]s are super-Turing, i.e.*
  Ev-RNN[$\mathbb{R}$]s are equivalent to TM/poly(A) in poly time.

Summary of the results

	Static	Evolving
$\mathbb{Q}$	Turing Siegelmann & Sontag 95	Super-Turing Cabessa & Siegelmann 11
$\mathbb{R}$	Super-Turing Siegelmann & Sontag 94	Super-Turing Cabessa & Siegelmann 11

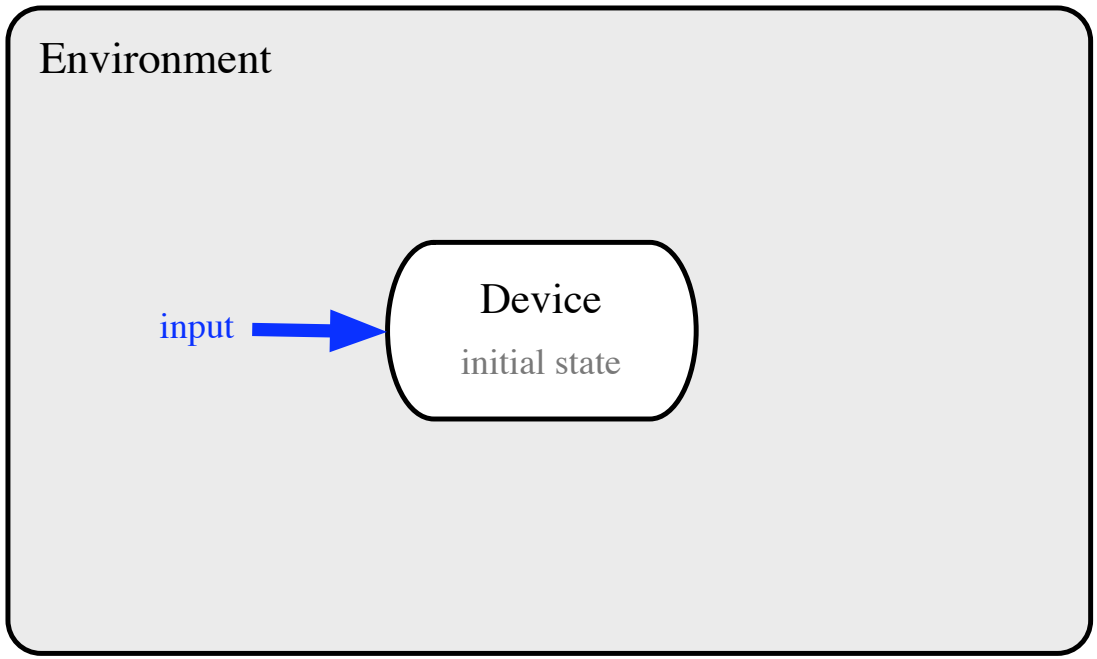
Classical computation



Closed-box and amnesic...

"[...] no longer fully corresponds to the current notion of computing in modern systems." (Van Leeuwen & Wiedermann 2008)

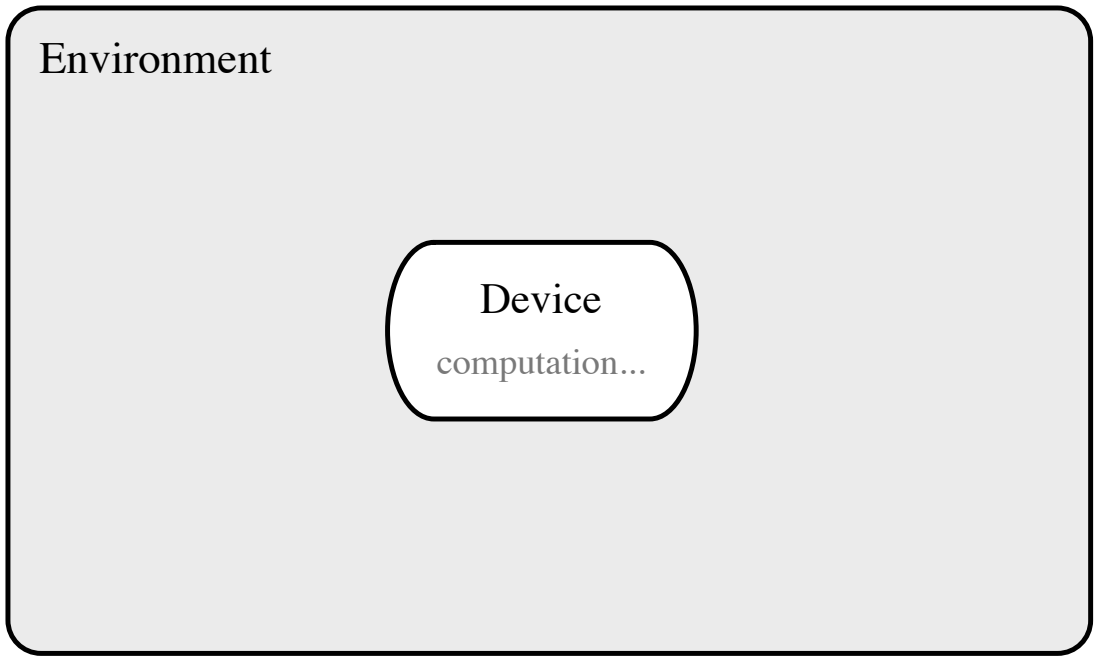
Classical computation



Closed-box and amnesic...

"[...] no longer fully corresponds to the current notion of computing in modern systems." (Van Leeuwen & Wiedermann 2008)

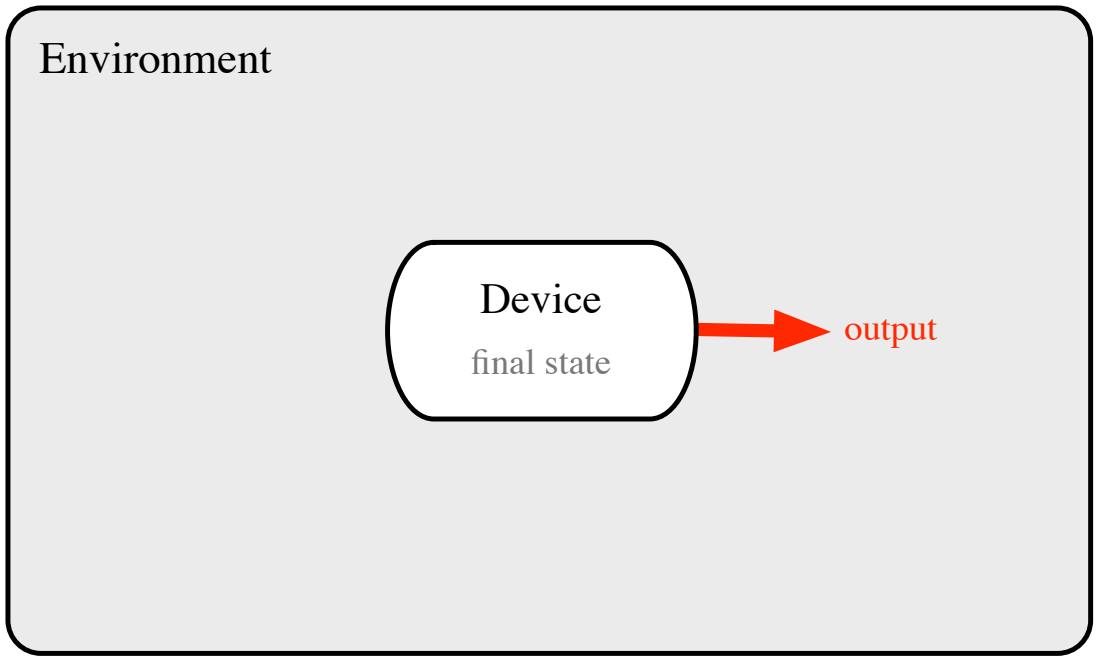
Classical computation



Closed-box and amnesic...

"[...] no longer fully corresponds to the current notion of computing in modern systems." (Van Leeuwen & Wiedermann 2008)

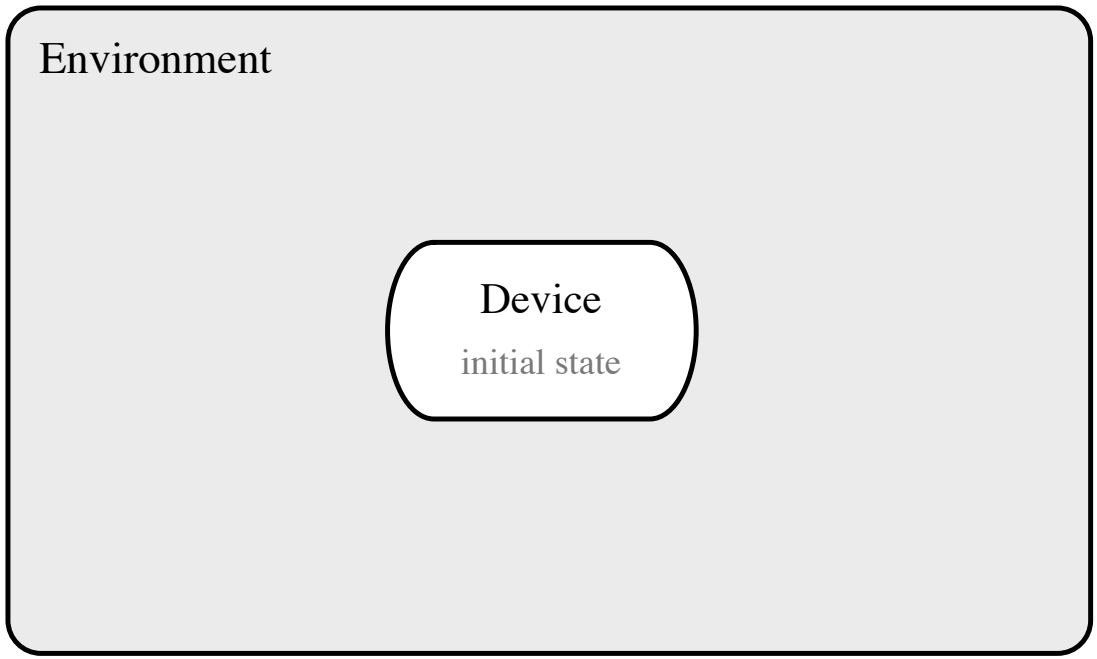
Classical computation



Closed-box and amnesic...

"[...] no longer fully corresponds to the current notion of computing in modern systems." (Van Leeuwen & Wiedermann 2008)

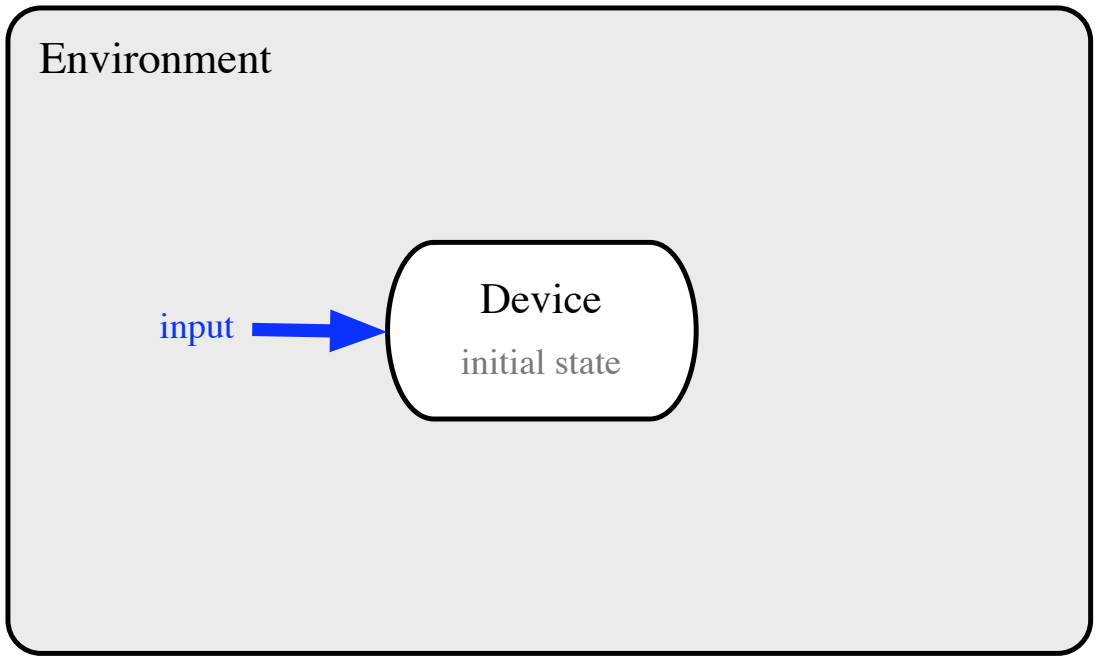
Classical computation



Closed-box and amnesic...

"[...] no longer fully corresponds to the current notion of computing in modern systems." (Van Leeuwen & Wiedermann 2008)

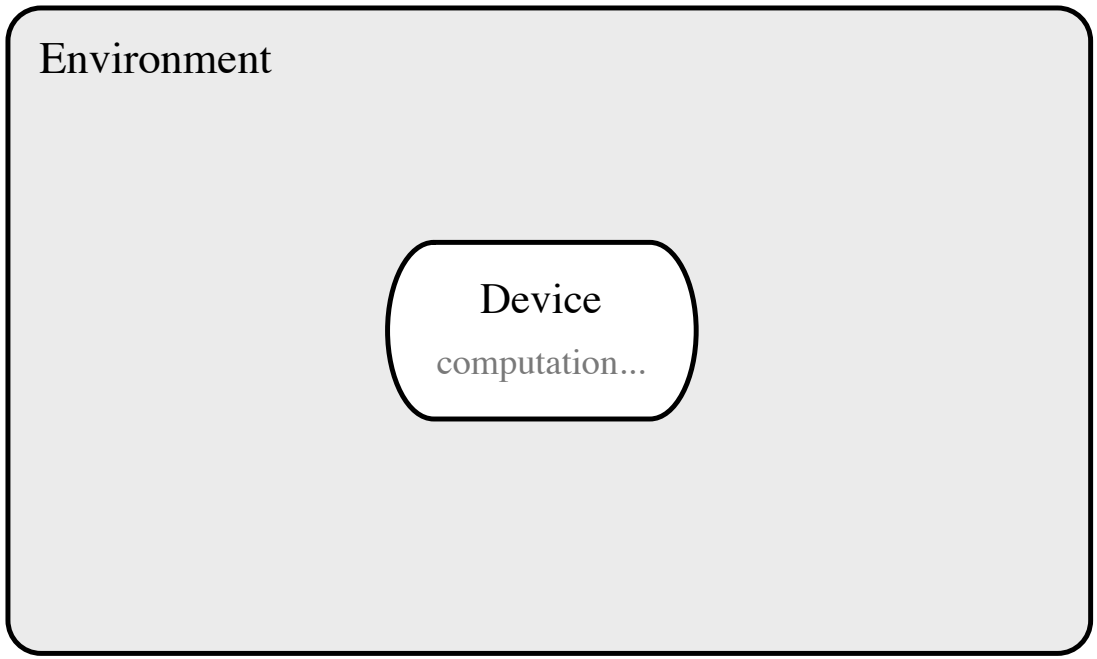
Classical computation



Closed-box and amnesic...

"[...] no longer fully corresponds to the current notion of computing in modern systems." (Van Leeuwen & Wiedermann 2008)

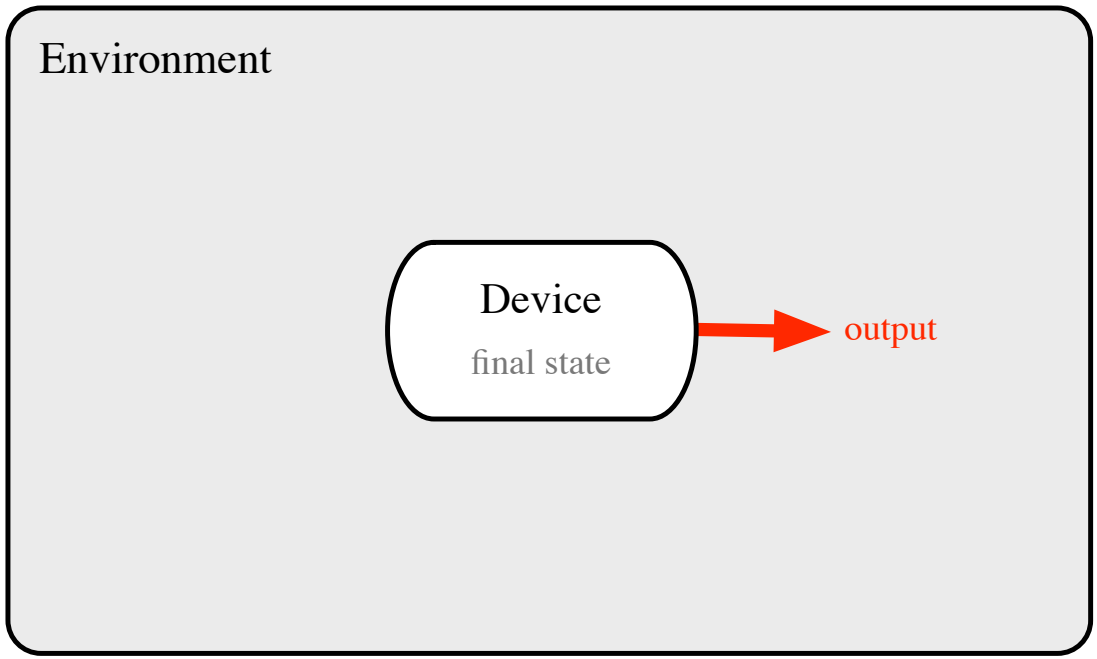
Classical computation



Closed-box and amnesic...

"[...] no longer fully corresponds to the current notion of computing in modern systems." (Van Leeuwen & Wiedermann 2008)

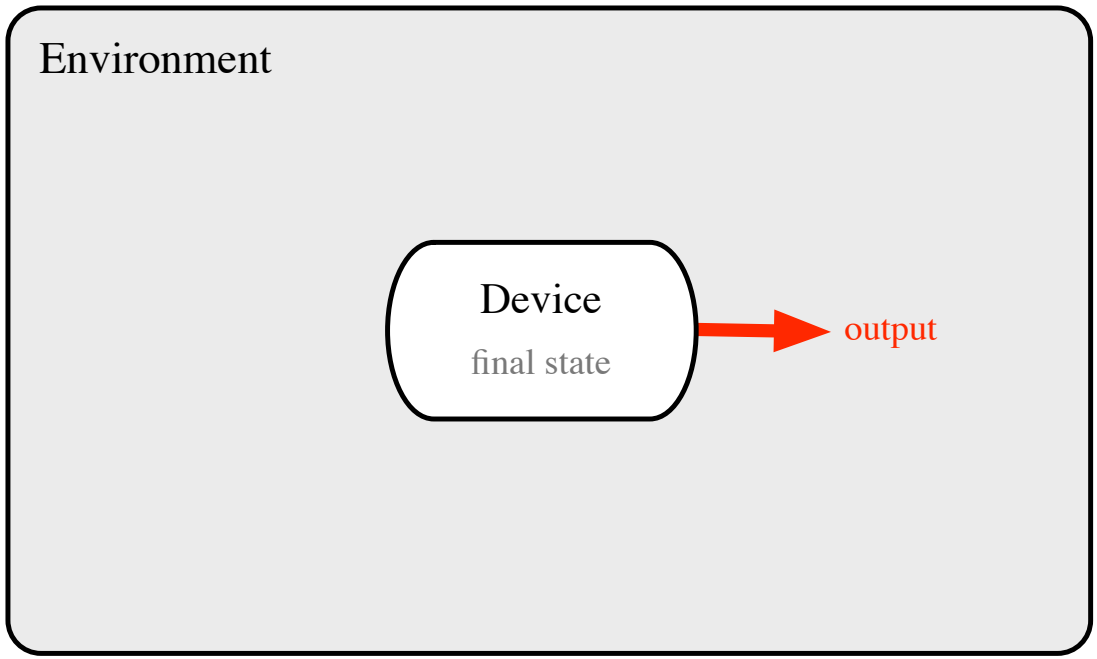
Classical computation



Closed-box and amnesic...

"[...] no longer fully corresponds to the current notion of computing in modern systems." (Van Leeuwen & Wiedermann 2008)

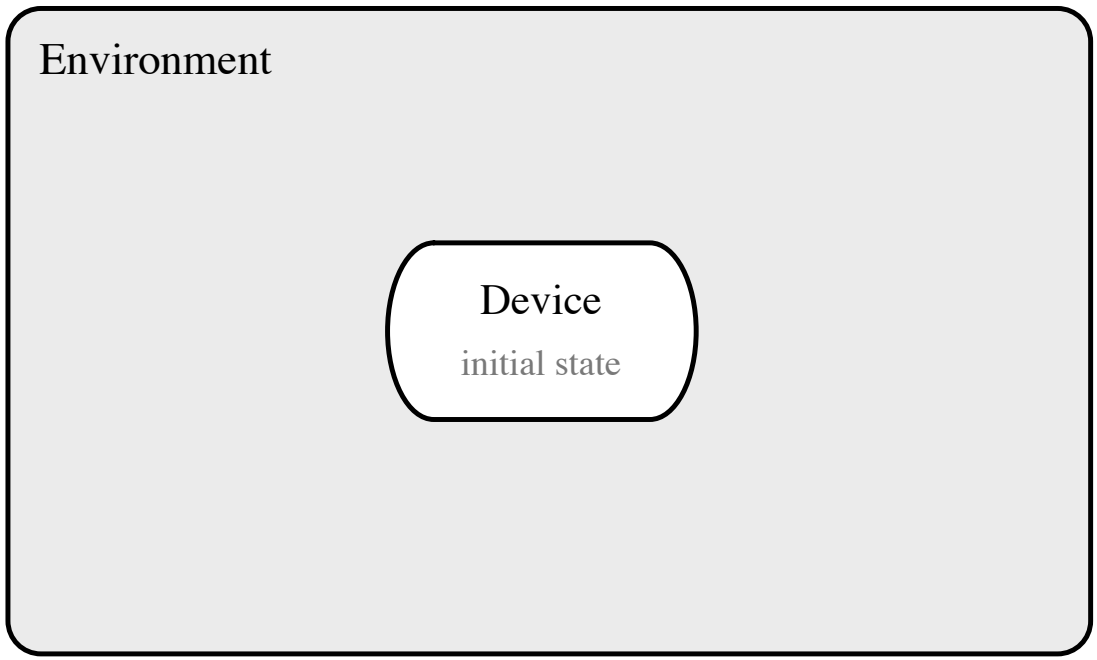
Classical computation



Closed-box and amnesic...

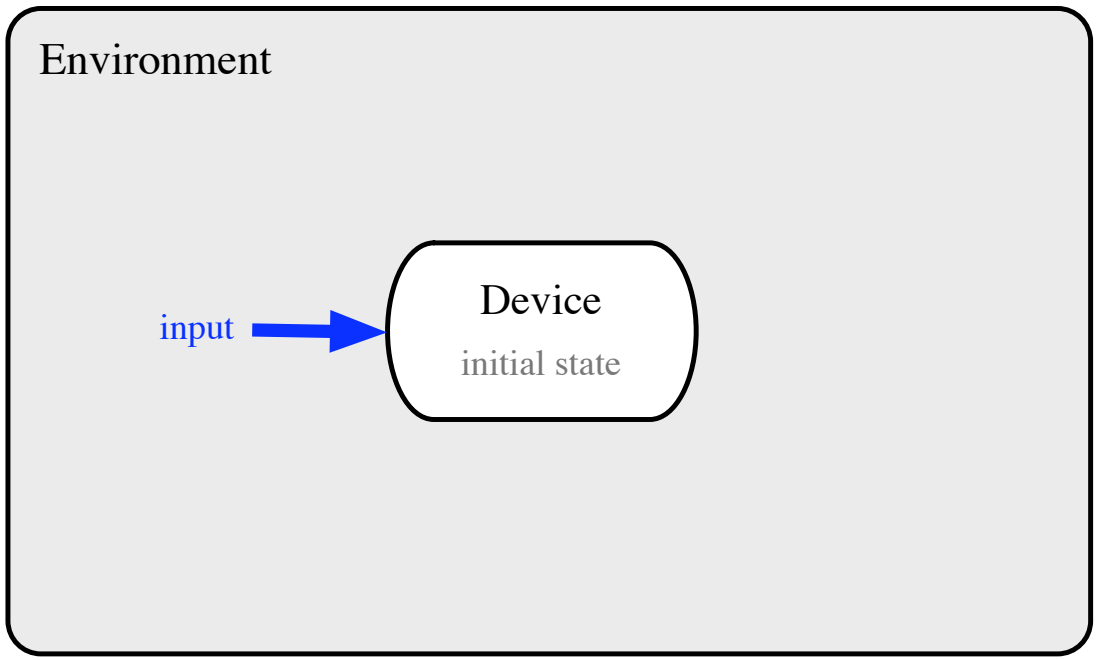
> *"[...] no longer fully corresponds to the current notion of computing in modern systems." (Van Leeuwen & Wiedermann 2008)*

Interactive computation



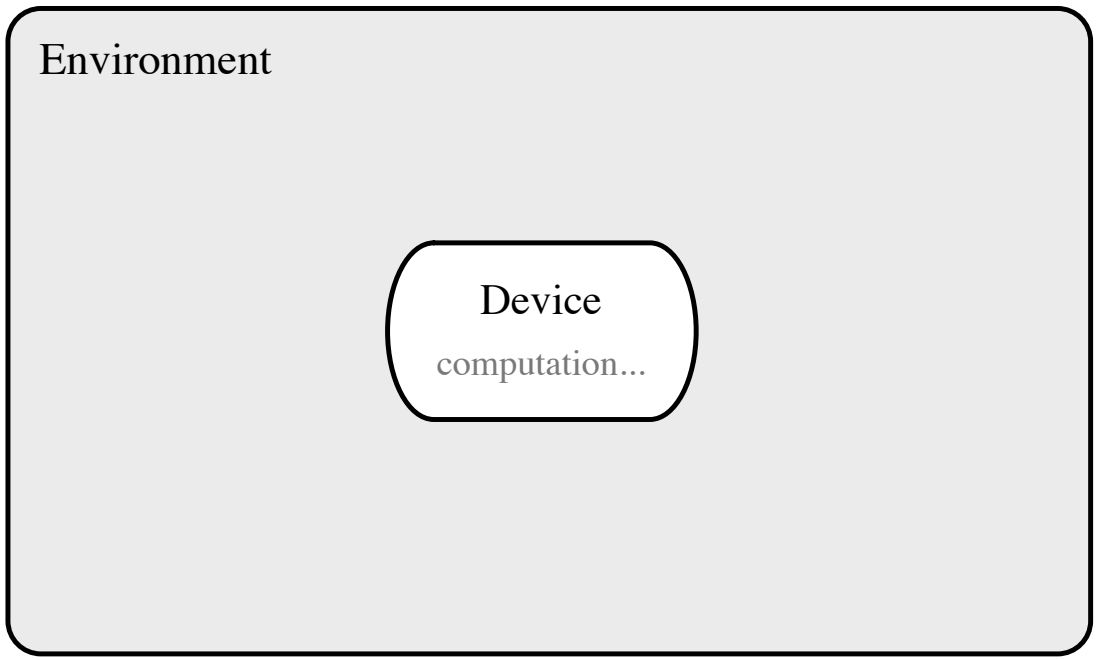
Sequentially interactive and memory active... More appropriate for bio-inspired complex information processing systems.

Interactive computation



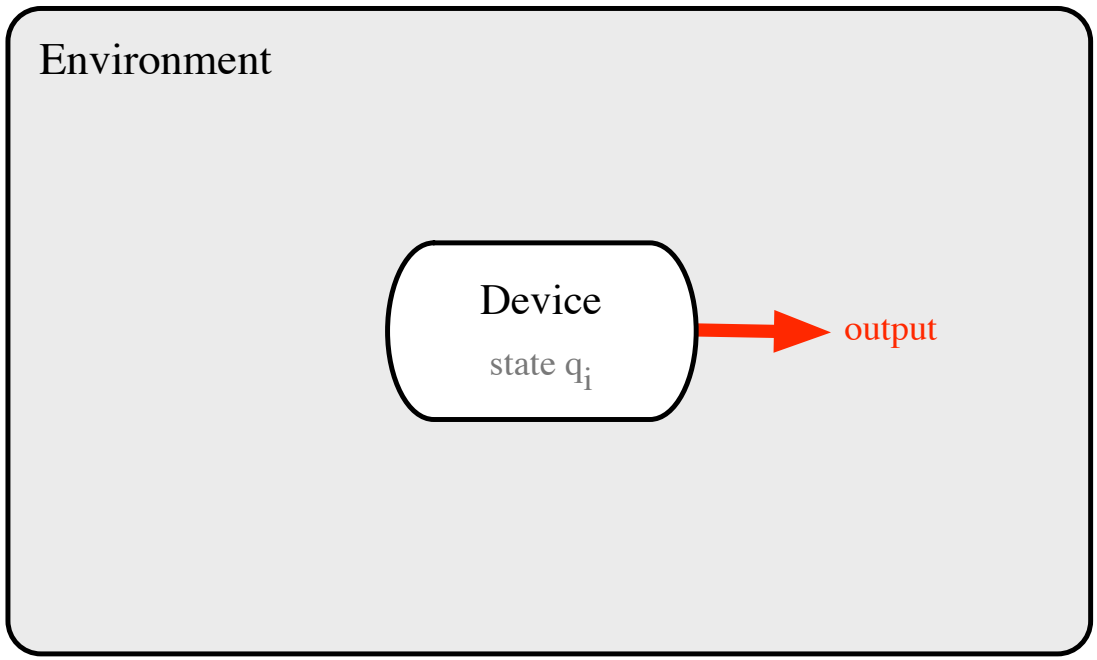
Sequentially interactive and memory active... More appropriate for bio-inspired complex information processing systems.

Interactive computation



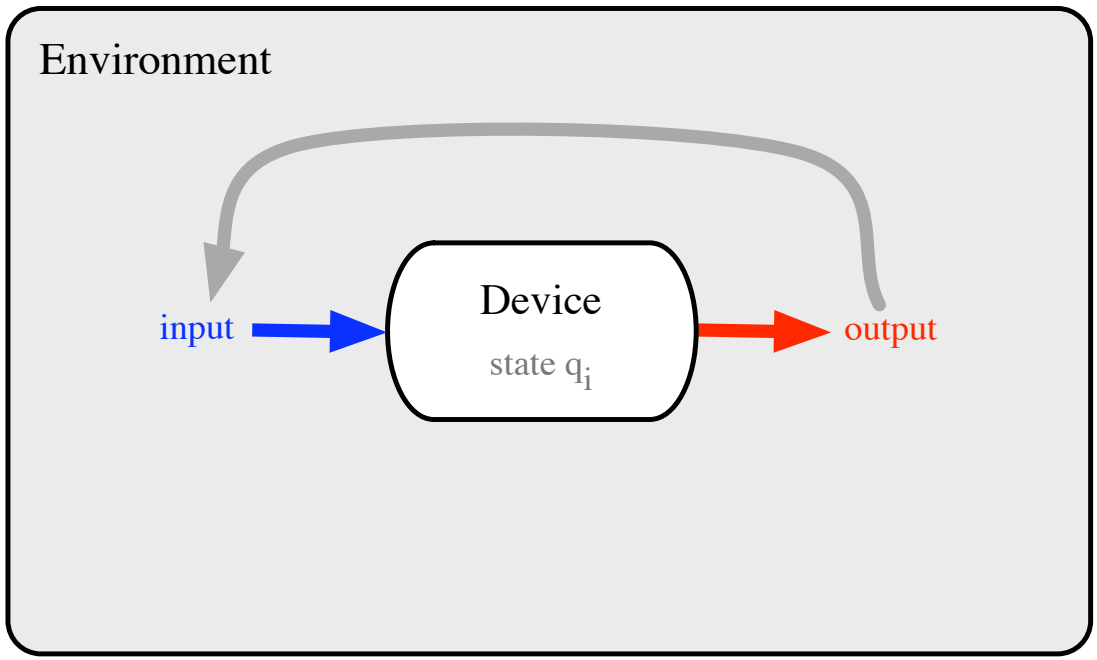
Sequentially interactive and memory active... More appropriate for bio-inspired complex information processing systems.

Interactive computation



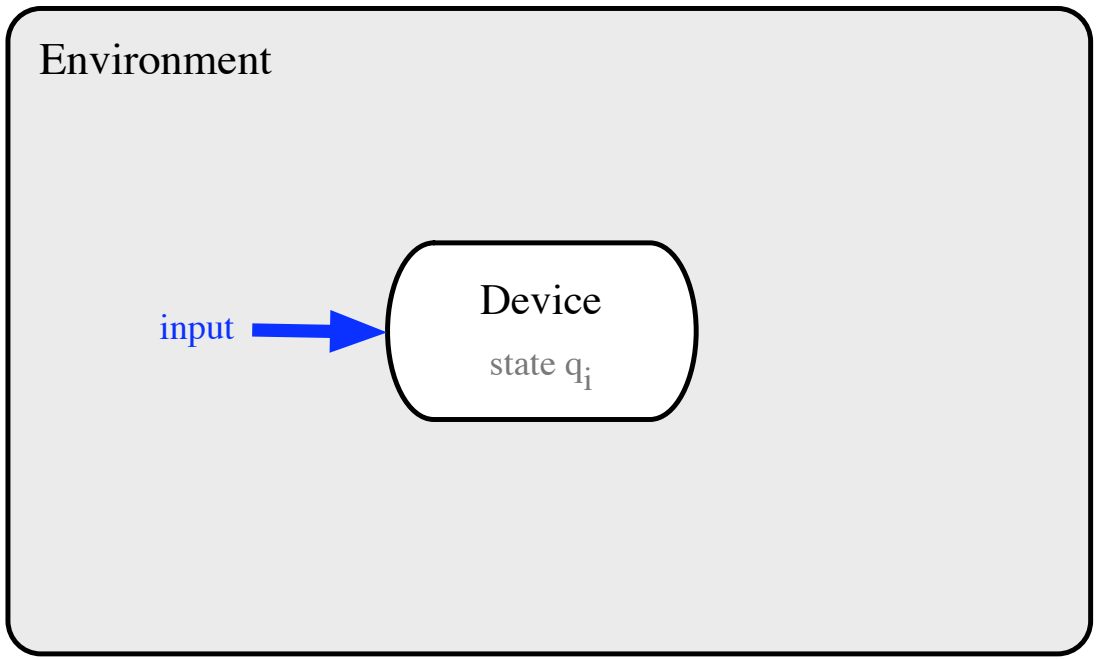
Sequentially interactive and memory active... More appropriate for bio-inspired complex information processing systems.

Interactive computation



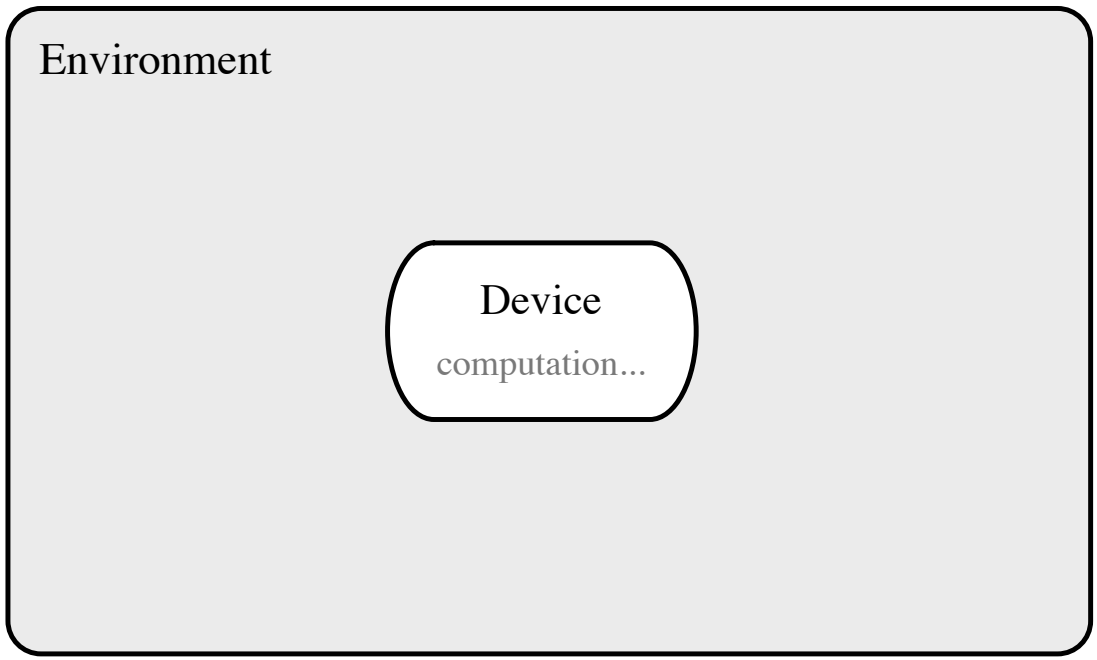
Sequentially interactive and memory active... More appropriate for bio-inspired complex information processing systems.

Interactive computation



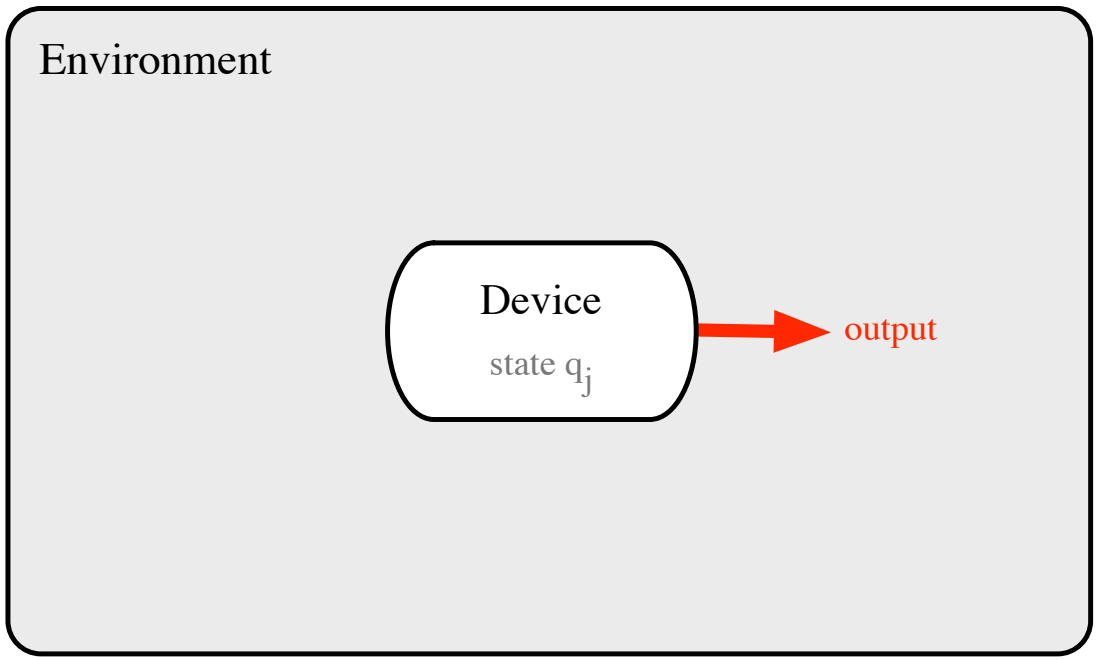
Sequentially interactive and memory active... More appropriate for bio-inspired complex information processing systems.

Interactive computation



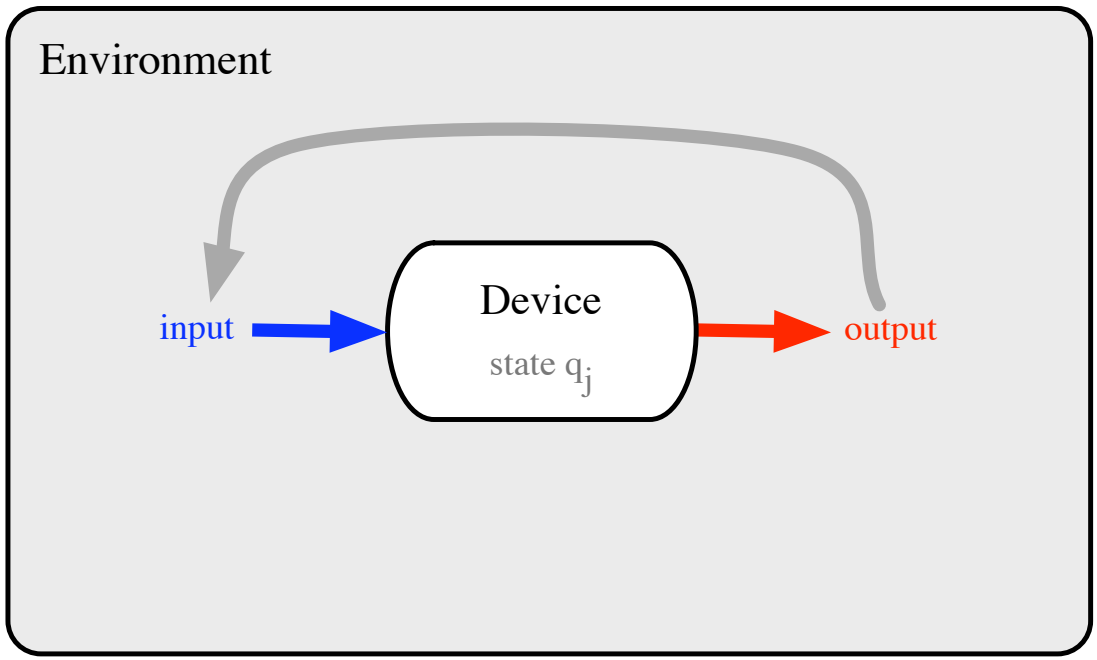
Sequentially interactive and memory active... More appropriate for bio-inspired complex information processing systems.

Interactive computation



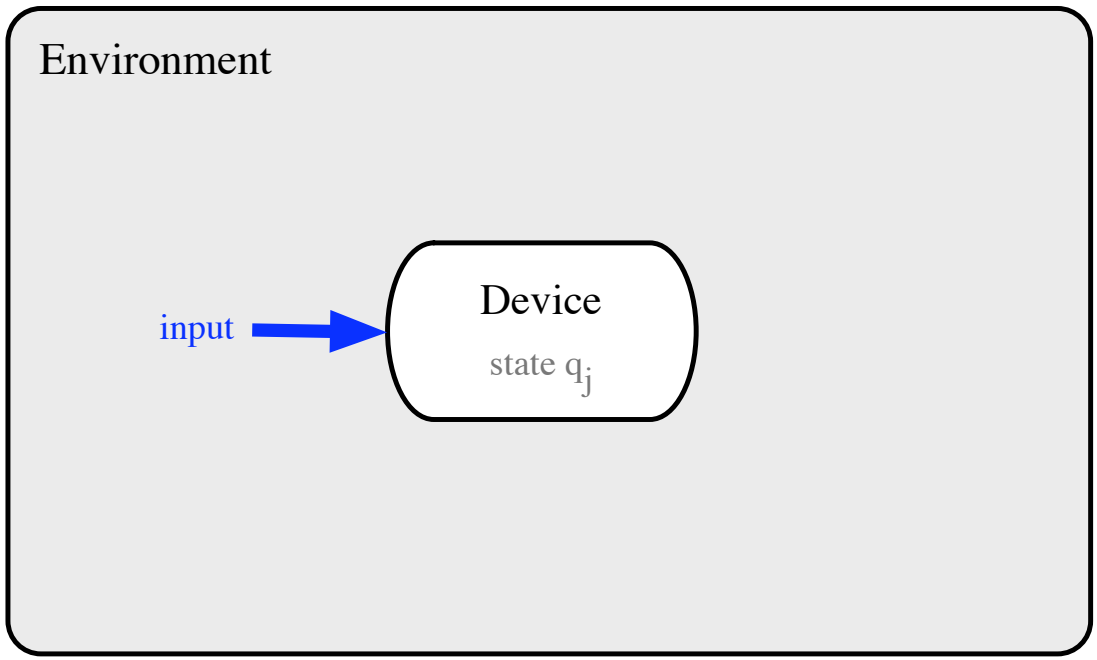
Sequentially interactive and memory active... More appropriate for bio-inspired complex information processing systems.

Interactive computation



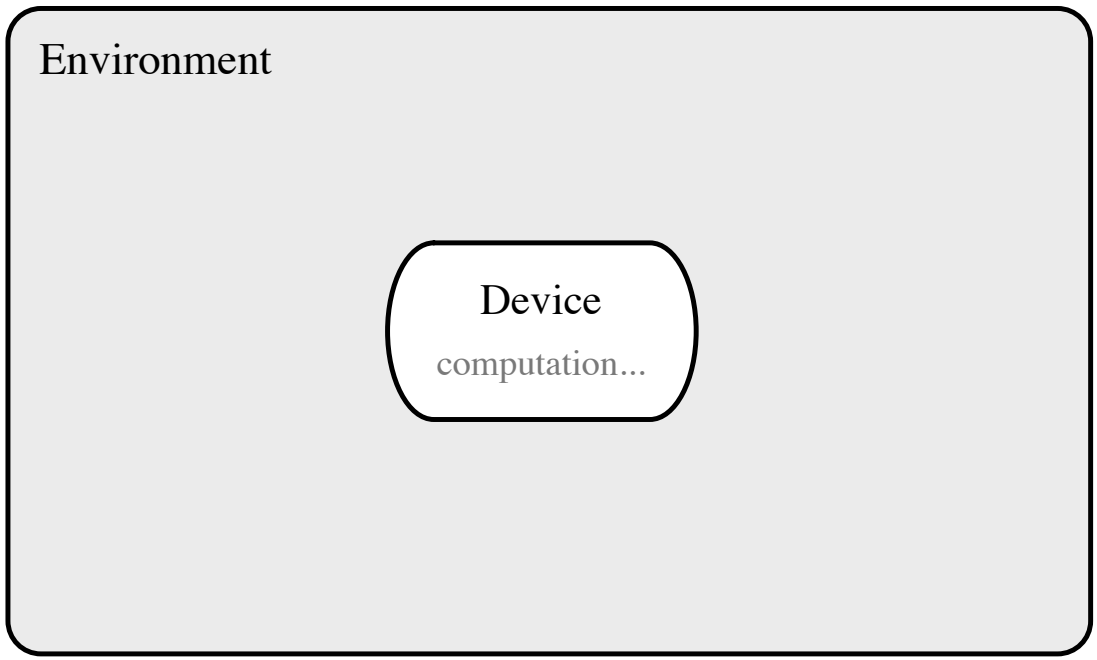
Sequentially interactive and memory active... More appropriate for bio-inspired complex information processing systems.

Interactive computation



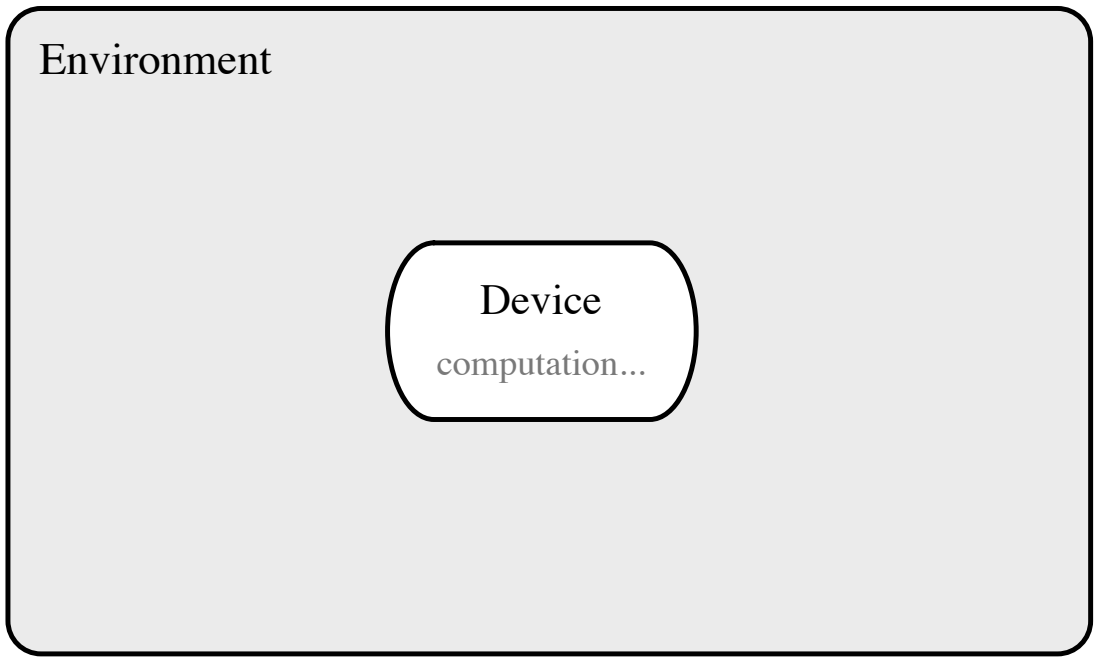
Sequentially interactive and memory active... More appropriate for bio-inspired complex information processing systems.

Interactive computation



Sequentially interactive and memory active... More appropriate for bio-inspired complex information processing systems.

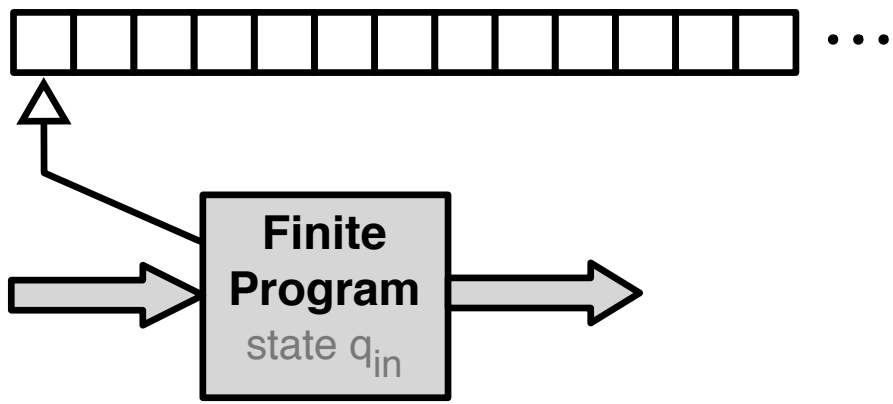
Interactive computation



Sequentially interactive and memory active... More appropriate for bio-inspired complex information processing systems.

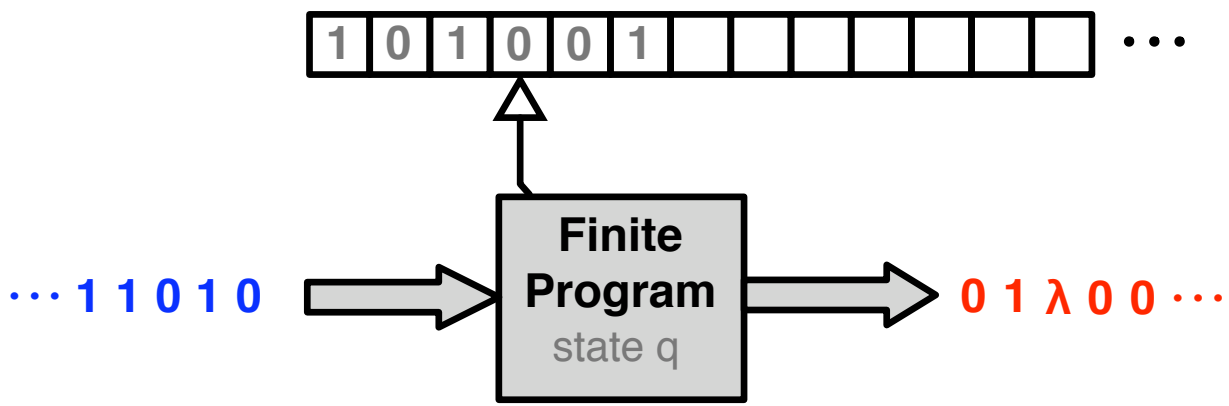
Interactive Turing machine

An interactive Turing machine (Int-TM) consists of a work tape, an input and an output port, a read-write head, and a finite program.



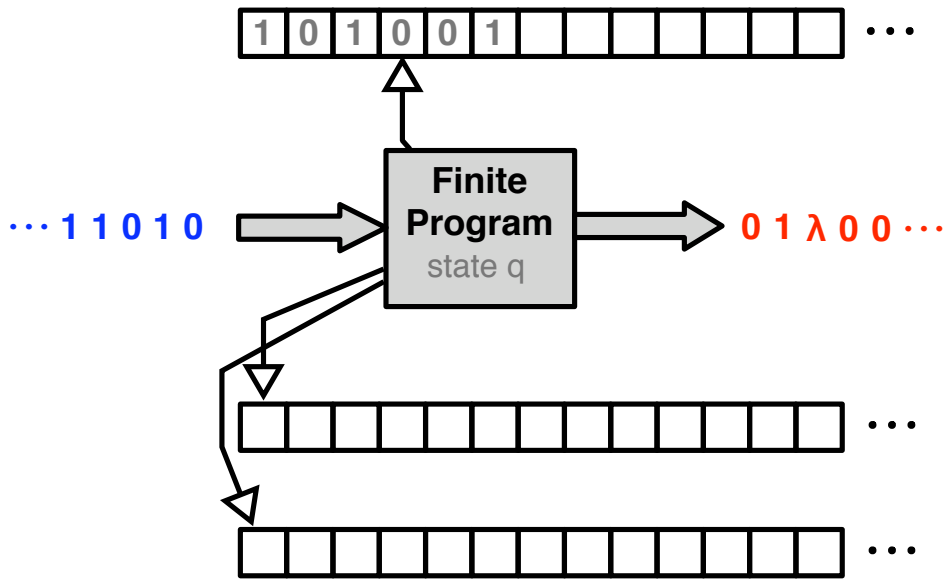
Interactive Turing machine

An interactive Turing machine (Int-TM) consists of a work tape, an input and an output port, a read-write head, and a finite program.



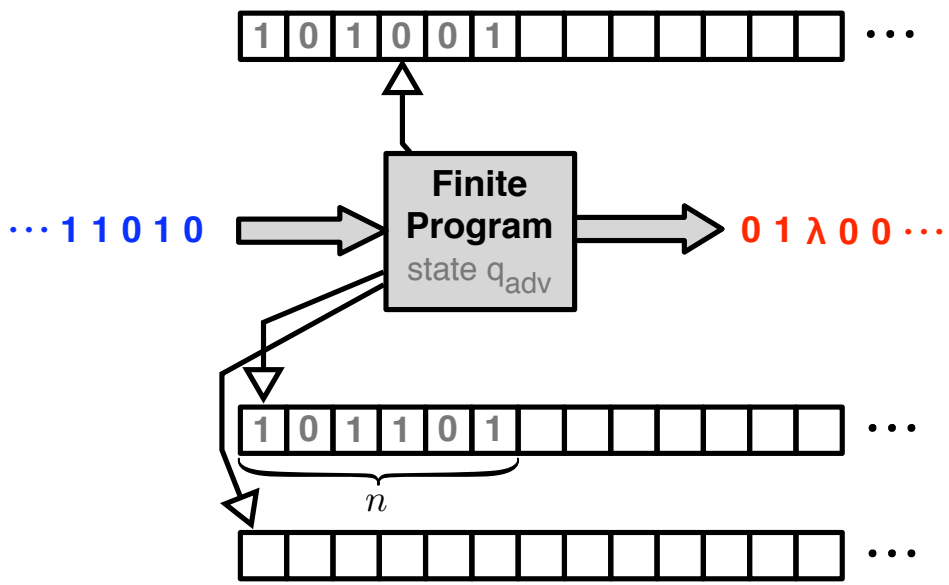
Interactive Turing machine with advice

An interactive TM with advice (Int-TM/A) is an Int-TM provided with additional advice input and output tapes and advice function $\alpha : \mathbb{N} \longrightarrow \{0,1\}^*$.



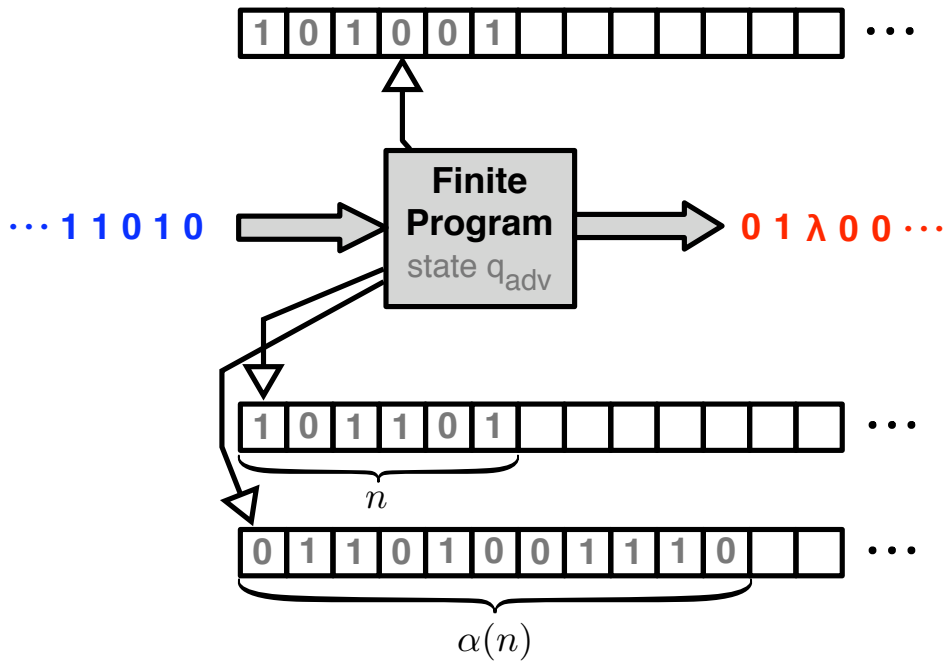
Interactive Turing machine with advice

An interactive TM with advice (Int-TM/A) is an Int-TM provided with additional advice input and output tapes and advice function $\alpha : \mathbb{N} \longrightarrow \{0,1\}^*$.



Interactive Turing machine with advice

An interactive TM with advice (Int-TM/A) is an Int-TM provided with additional advice input and output tapes and advice function $\alpha : \mathbb{N} \longrightarrow \{0,1\}^*$.



Lemma

Int-TM/As are strictly more powerful than int-TMs, i.e., they can compute strictly more ω-translations.

Computational power

The results concerning the computational power of RNNs in classical computation generalize to the interactive computational framework.

	Static	Evolving
$\mathbb{Q}$	Turing Cabessa & Siegelmann 12	Super-Turing Cabessa 12
$\mathbb{R}$	Super-Turing Cabessa & Siegelmann 12 Cabessa & Villa 12	Super-Turing Cabessa 12

Conclusions

- Recurrent neural networks provide a natural abstract model of computation beyond the Turing limits (Cabessa & Villa 12).
- *Architectural Evolution* represents an equivalent alternative to the *power of the continuum* towards the achievement of super-Turing capabilities.
- The results support the idea that *architectural evolution* might play a crucial role in the computational capabilities of biological neural networks.
- Future work: study the computational power of other kinds of architecturally evolving networks: other sigmoid functions, other learning rules, spiking networks, etc.

Conclusions

- Recurrent neural networks provide a natural abstract model of computation beyond the Turing limits (Cabessa & Villa 12).
- *Architectural Evolution* represents an equivalent alternative to the *power of the continuum* towards the achievement of super-Turing capabilities.
- The results support the idea that *architectural evolution* might play a crucial role in the computational capabilities of biological neural networks.
- Future work: study the computational power of other kinds of architecturally evolving networks: other sigmoid functions, other learning rules, spiking networks, etc.

Conclusions

- Recurrent neural networks provide a natural abstract model of computation beyond the Turing limits (Cabessa & Villa 12).
- *Architectural Evolution* represents an equivalent alternative to the *power of the continuum* towards the achievement of super-Turing capabilities.
- The results support the idea that *architectural evolution* might play a crucial role in the computational capabilities of biological neural networks.
- Future work: study the computational power of other kinds of architecturally evolving networks: other sigmoid functions, other learning rules, spiking networks, etc.

Conclusions

- Recurrent neural networks provide a natural abstract model of computation beyond the Turing limits (Cabessa & Villa 12).
- *Architectural Evolution* represents an equivalent alternative to the *power of the continuum* towards the achievement of super-Turing capabilities.
- The results support the idea that *architectural evolution* might play a crucial role in the computational capabilities of biological neural networks.
- Future work: study the computational power of other kinds of architecturally evolving networks: other sigmoid functions, other learning rules, spiking networks, etc.

Bibliography

Cabessa, J. (2012). Interactive evolving recurrent neural networks are super-Turing. In Filipe, J. and Fred, A., editors, *ICAART 2012: Proceedings of the 4th International Conference on Agents and Artificial Intelligence 2012*, volume 1, pages 328–333. SciTePress.

Cabessa, J. and Siegelmann, H. T. (2011). Evolving recurrent neural networks are super-Turing. In *IJCNN 2011: Proceedings of the International Joint Conference on Neural Networks 2011*, pages 3200–3206. IEEE.

Cabessa, J. and Villa, A. E. (2012a). The expressive power of analog recurrent neural networks on infinite input streams. *Theor. Comput. Sci.*, 436:23–34.

Cabessa, J. and Villa, A. E. (2012b). Recurrent neural networks: A natural model of computation beyond the Turing limits. In *NCTA 2012: Proceedings of the 4th International Conference on Neural Computation Theory and Applications*, pages accepted, to appear. SciTePress.

Siegelmann, H. T. and Sontag, E. D. (1994). Analog computation via neural networks. *Theor. Comput. Sci.*, 131(2):331–360.

Siegelmann, H. T. and Sontag, E. D. (1995). On the computational power of neural nets. *J. Comput. Syst. Sci.*, 50(1):132–150.