Computational Capabilities of Recurrent Neural Networks Based on their Attractor Dynamics

Jérémie Cabessa
Department of Mathematical Economics
University Panthéon-Assas – Paris 2
75006 Paris, France
Email: jeremie.cabessa@u-paris2.fr

Alessandro E.P. Villa
Department of Information Systems
University of Lausanne
CH-1015 Lausanne, Switzerland
Email: alessandro.villa@unil.ch

***Abstract*—We consider a model of so-called hybrid recurrent neural networks composed with Boolean input and output cells as well as sigmoid internal cells. When subjected to some infinite binary input stream, the Boolean output cells necessarily exhibit some attractor dynamics, which is assumed to be of two possible kinds, namely either meaningful or spurious, and which underlies the arising of spatiotemporal patterns of output discharges. In this context, we show that rational-weighted neural networks are computationally equivalent to deterministic Muller Turing machines, whereas all other models of real-weighted or evolving neural networks are equivalent to each other, and strictly more powerful than deterministic Muller Turing machines. In this precise sense, the analog and evolving neural networks are super-Turing. We further provide some precise mathematical characterization of the expressive powers of all these neural models. These results constitute a generalization to the current computational context of those obtained in the cases of classical as well as interactive computations. They support the idea that recurrent neural networks represent a natural model of computation beyond the Turing limits.**

I. Introduction

In neural computation, understanding the computational and dynamical capabilities of brain-like models represents an issue of central importance. In this context, much attention has been focused on comparing the computational capabilities of various neural models to those of diverse abstract machines, see [1]–[20]. As a consequence, the computational power of neural networks has been shown to be intimately related to the nature of their synaptic weights and activation functions, and capable to range from finite state automata up to super-Turing capabilities.

Besides, the hypothesis that neuronal information is processed in time both individually and jointly following precise time intervals was initially postulated when the nervous system was conceptualized as dynamic networks of interacting neurons [1], and experimentally demonstrated few years later [21]. The combined activity in the neurons that are afferent to a cell are necessaryily affecting its activity. Re-entrant activity is likely to occur in most brain circuits due to the existence of recurrent connections within most biological neural networks. Thus, certain pathways within the networks are likely to be potentiated or weakened by developmental and/or learning processes, by affecting the number or the efficacy of synaptic interactions between the neurons. It is rationale to suppose that at short time scale, despite the plasticity of these phenomena, the same input information presented in the network is evoking very similar, if not identical, patterns of activity in a cell assembly, referred here as a small circuit of functionally interconnected neurons. Such recurring, ordered, and precise patterns of activity correspond to ordered sequences of interspike intervals, referred to as spatiotemporal patterns of discharges or preferred firing sequences. Notably, the arising of such spatiotemporal patterns of discharges is assumed to be related to the attractor dynamics of the neural networks.

Following this global line of thought, Cabessa and Villa initiated the study of the expressive power of recurrent neural networks from the perspective of their attractor dynamics [12]–[15]. They proved that Boolean recurrent neural networks provided with some assignment of their attractors into two different kinds are computationally equivalent to Muller automata, and hence recognize precisely the so-called ω-regular neural languages. Consequently, the most refined topological classification of ω-languages can be transposed from the automaton to the neural network context, and yield to some transfinite hierarchical classification of Boolean neural network according to their attractor dynamics, which in turn represents a new attractor-based complexity measurement for Boolean recurrent neural networks [15].

Here, this precise research direction is pursued. More precisely, we consider a model of so-called *hybrid recurrent neural networks* composed with Boolean input and output cells as well as sigmoid internal cells. When subjected to some infinite binary input stream, the Boolean output cells necessarily exhibit some attractor dynamics, which is assumed to be of two possible kinds, namely either meaningful or spurious, and which underlies the arising of spatiotemporal patterns of output discharges. In this context, we show that rational-weighted hybrid neural networks are computationally equivalent to deterministic Muller Turing machines, whereas all other models of real-weighted or evolving hybrid networks are equivalent to each other, and strictly more powerful than deterministic Muller Turing machines. In this precise sense, the analog and evolving hybrid neural networks are *super-Turing*. These results provide a precise generalization to the current computational context of those obtained in the cases of classical and interactive computations [10], [11], [16], [18]–[20]. They further provide a step forward in the study of the computational capabilities of brain-like models.

978-1-4799-1959-8/15/$31.00

II. Preliminaries

For any $k > 0$, the space of k-dimensional Boolean vectors is denoted by $\mathbb{B}^k$. The spaces of finite and infinite sequences of k-dimensional Boolean vectors are denoted by $(\mathbb{B}^k)^*$ and $(\mathbb{B}^k)^\omega$, respectively. Any finite sequence $s \in (\mathbb{B}^k)^*$ of length n will be denoted by an expression of the form $s = \vec{s}(0) \cdots \vec{s}(n-1)$, and any infinite sequence $s \in (\mathbb{B}^k)^\omega$ will be denoted by $s = \vec{s}(0)\vec{s}(1)\vec{s}(2)\cdots$, where each $\vec{s}(i) \in \mathbb{B}^k$. If s has length more than n, we let $s[0{:}n]$ denote the sequence consisting of the n first elements of s, with the convention that $s[0{:}0]$ is the empty word. Moreover, for any $x \in (\mathbb{B}^k)^*$ and $y \in (\mathbb{B}^k)^* \cup (\mathbb{B}^k)^\omega$, the fact that x is a prefix (resp. strict prefix) of y will be denoted by $x \subseteq y$ (resp. $x \subsetneq y$). Then, for any finite sequence $p \in (\mathbb{B}^k)^*$, we set $p \cdot (\mathbb{B}^k)^\omega = \{x \in (\mathbb{B}^k)^\omega : p \subsetneq x\}$. In words, $p \cdot (\mathbb{B}^k)^*$ is the set of infinite sequences which contain p as a prefix.

As for the case of infinite words of bits [22], the space $(\mathbb{B}^k)^\omega$ can naturally be equipped with the product topology of the discrete topology on $\mathbb{B}^k$. Accordingly, the *basic open sets* of $(\mathbb{B}^k)^\omega$ are the sets of the form $p \cdot (\mathbb{B}^k)^\omega$, for some prefix $p \in (\mathbb{B}^k)^*$, and the *general open sets* of $(\mathbb{B}^k)^\omega$ are the countable unions of basic open sets, namely the sets of the form $\bigcup_{i \in I} p_i \cdot (\mathbb{B}^k)^\omega$, where $I \subseteq \mathbb{N}$ and each $p_i \in (\mathbb{B}^k)^*$.

The class of *Borel* subsets of $(\mathbb{B}^k)^\omega$ consists of the smallest collection of subsets of $(\mathbb{B}^k)^\omega$ containing all open sets and closed under countable union and complementation. The two first levels $\mathbf{\Sigma_1^0}$ and $\mathbf{\Pi_1^0}$ of the *Borel hierarchy* consist of the collections of all open and closed sets, namely:

$$\mathbf{\Sigma_1^0} = \{X \subseteq (\mathbb{B}^k)^\omega : X \text{ is open}\}$$

$$\mathbf{\Pi_1^0} = \{X \subseteq (\mathbb{B}^k)^\omega : X^c \in \mathbf{\Sigma_1^0}\}$$

The two second levels $\mathbf{\Sigma_2^0}$ and $\mathbf{\Pi_2^0}$ of the Borel hierarchy are the collections of countable unions of closed sets and countable intersections of open sets, i.e.:

$$\mathbf{\Sigma_2^0} = \{X \subseteq (\mathbb{B}^k)^\omega : X = \bigcup_{n \in \mathbb{N}} X_n,\ X_n \in \mathbf{\Pi_1^0}\}$$

$$\mathbf{\Pi_2^0} = \{X \subseteq (\mathbb{B}^k)^\omega : X = \bigcap_{n \in \mathbb{N}} X_n,\ X_n \in \mathbf{\Sigma_1^0}\}$$

In this context, any $\mathbf{\Pi_2^0}$-set X of $(\mathbb{B}^k)^\omega$ can be written in the form

$$X = \bigcap_{i \geq 0} \bigcup_{j \geq 0} p_{i,j} \cdot (\mathbb{B}^k)^\omega,$$

where each $p_{i,j} \in (\mathbb{B}^k)^*$. The *Boolean combinations* of $\mathbf{\Pi_2^0}$-sets, denoted by $BC(\mathbf{\Pi_2^0})$, is the closure of the class $\mathbf{\Pi_2^0}$ by finite Boolean operations on sets (i.e. union, intersection, and complementation). The class $BC(\mathbf{\Sigma_2^0})$ is defined analogously, and one clearly has $BC(\mathbf{\Pi_2^0}) = BC(\mathbf{\Sigma_2^0})$.

Besides, the space of infinite sequence of bits is denoted by $\{0,1\}^\omega$, and an element $w \in \{0,1\}^\omega$ is denoted by $w = w(0)w(1)w(2)\cdots$, where each $w(i) \in \{0,1\}$. The space $\{0,1\}^\omega$ can also be equipped with the product topology of the discrete topology on $\{0,1\}$, and the collection of open, closed, and Borel sets can naturally be defined. In this infinite word context, a *Muller Turing machine* $\mathcal{M}$ consists of a Turing machine (TM) equipped with a so-called Muller table $\mathcal{T} = \{T_1, \ldots, T_k\}$, where each T_i is a subset of the states of $\mathcal{M}$. An infinite word $w \in \{0,1\}^\omega$ is said to be *recognized* by $\mathcal{M}$ if the set of states visited infinitely often by $\mathcal{M}$ during the processing of w belongs to its table $\mathcal{T}$. The *language recognized by* $\mathcal{M}$, denoted by $L(\mathcal{M})$, consists of the set of all words recognized by $\mathcal{M}$. It is known that any language recognized by some deterministic Muller Turing machine belongs to the class $BC(\mathbf{\Pi_2^0})$ of $\{0,1\}^\omega$ [22]. However, a simple cardinality argument shows that not all $BC(\mathbf{\Pi_2^0})$-sets of $\{0,1\}^\omega$ can be recognized by some deterministic Muller Turing machine.[1] These results can be directly transposed in the context of Turing machines working on $(\mathbb{B}^k)^\omega$ instead of $\{0,1\}^\omega$. Finally, in the sequel, any function or procedure that can be computed or performed by some Turing machine will be called *recursive*.

III. The Model

We introduce a model of so-called *hybrid recurrent neural network* which involves the consideration of both Boolean and sigmoid cells, and whose output significance is related to the attractor dynamics of its Boolean output cells.

A *hybrid (or Boolean/sigmoid) recurrent neural network* (B/S-RNN) consists of a synchronous network of neurons related together in a general architecture. The network contains N internal sigmoid neurons $(x_i)_{i=1}^N$, M Boolean input cells $(u_i)_{i=1}^M$, and P Boolean output cells $(y_i)_{i=1}^P$. The dynamics of the network is computed as follows: given the activation values of the input and internal neurons $(u_j)_{j=1}^M$ and $(x_j)_{j=1}^N$ at time t, the activation values of each internal neuron x_i and each output neuron y_i at time $t+1$ are updated by the following equations, respectively

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

$$y_i(t+1) = \theta\left(\sum_{j=1}^{N} a_{ij}(t) \cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t) \cdot u_j(t) + c_i(t)\right), \quad \text{for } i = 1, \ldots, P \quad (2)$$

where the $a_{ij}(t)$, $b_{ij}(t)$, and $c_i(t)$ are time dependent values describing the weighted synaptic connections and weighted bias of the network, σ is the classical saturated-linear activation function defined by

$$\sigma(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } 0 \leq x \leq 1 \\ 1 & \text{if } x > 1 \end{cases}$$

and θ is the hard-threshold function defined by

$$\theta(x) = \begin{cases} 0 & \text{if } x < 1 \\ 1 & \text{if } x \geq 1 \end{cases}$$

[1] Indeed, there are $\aleph_0$ Muller Turing machines and $2^{\aleph_0}$ sets in $BC(\mathbf{\Pi_2^0})$.

Hence, the dynamics of any B/S-RNN $\mathcal{N}$ is given by the function $f_{\mathcal{N}} : \mathbb{B}^M \times \mathbb{B}^N \to \mathbb{B}^N \times \mathbb{B}^P$ naturally defined by

$$f_{\mathcal{N}}(\vec{u}(t), \vec{x}(t)) = (\vec{x}(t+1), \vec{y}(t+1)). \tag{3}$$

where the components of $\vec{x}(t+1)$ and $\vec{y}(t+1)$ are given by Equations (1) and (2), respectively.

Throughout this paper, six different models of B/S-RNNs are considered according to whether the synaptic weights of the networks are modelled by rational or real numbers, and to whether these synaptic weights are either of a static nature, or able to evolve over time among only two possible values, or able to evolve over time among any possible values between two designated bounds s and s'.

Accordingly, a B/S-RNN will be called *rational* if its synaptic weights $a_{ij}(t), b_{ij}(t), c_i(t)$ are modelled by rational numbers, and *real* or *analog* if its weights are modelled by real numbers. It will be called *static* if its synaptic weights remains constant over time, *bi-valued evolving* if its synaptic weights might evolve among only two possible values over time (for instance 0 and 1), and *general evolving* (or simply *evolving*) if its synaptic weights might evolve among every possible values between two bounds s and s' imposed by the biological constitution of the synapses. In order to designate these six different neural models, the notations mentioned in Table I will be employed.

TABLE I: The six models of B/S-RNNs according to whether the synaptic weights of the network are modelled by rational or real numbers of either a static or a bi-valued evolving or a general evolving nature.

	STATIC	BI-VALUED EVOLVING	EVOLVING
$\mathbb{Q}$	B/S-RNN[$\mathbb{Q}$]	Ev$_2$-B/S-RNN[$\mathbb{Q}$]	Ev-B/S-RNN[$\mathbb{Q}$]
$\mathbb{R}$	B/S-RNN[$\mathbb{R}$]	Ev$_2$-B/S-RNN[$\mathbb{R}$]	Ev-B/S-RNN[$\mathbb{R}$]

Note that since rational numbers are real numbers, any rational B/S-RNN is a special case of a real B/S-RNN by definition. Moreover, since static synaptic weights are evolving weights that remain constant over time, any static B/S-RNN is a special case of an evolving (bi-valued of general) B/S-RNN. Also, any bi-valued evolving B/S-RNN is a special case of a general evolving B/S-RNN. According to these considerations, the relationships between the computational powers of the six models of B/S-RNNs illustrated in Figure 1 hold.

Consider some B/S-RNN $\mathcal{N}$ provided with N sigmoid cells, M Boolean input cells, and P Boolean output cells. For each time step $t \geq 0$, the Boolean vector

$$\vec{u}(t) = (u_1(t), \ldots, u_M(t)) \in \mathbb{B}^M$$

describing the spiking configurations of the input units of $\mathcal{N}$ at time t is the *input* submitted to $\mathcal{N}$ at time t. The pair

$$\langle \vec{x}(t), \vec{y}(t) \rangle \in [0,1]^N \times \mathbb{B}^P$$

describing the activation values and spiking configuration of the internal and output cells at time t is the *state* of $\mathcal{N}$ at time t. The second element of this pair, namely $\vec{y}(t)$, is the *Boolean state* of $\mathcal{N}$ at time t.

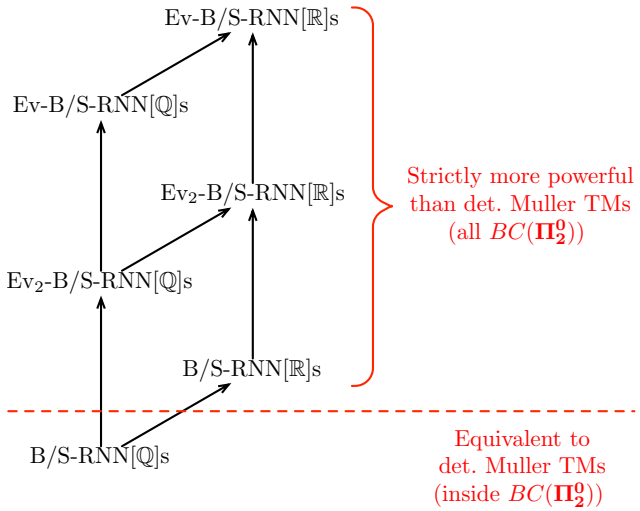

Fig. 1: Relationships between the computational powers of the six models of B/S-RNNs. There is an arrow from one model to the other if the former is less powerful than or equally powerful to the latter. The relation represented by these arrows is clearly transitive. In this paper, we show that B/S-RNN[$\mathbb{Q}$]s are computationally equivalent to deterministic Muller Turing machines (Theorem 1), and that the five other models of Ev$_2$-B/S-RNN[$\mathbb{Q}$]s, Ev-B/S-RNN[$\mathbb{Q}$]s, B/S-RNN[$\mathbb{R}$]s, Ev$_2$-B/S-RNN[$\mathbb{R}$]s, Ev-B/S-RNN[$\mathbb{R}$]s are equivalent to each other, and strictly more powerful than deterministic Muller Turing machines (Theorem 2).

Assuming the initial state of the network to be $\langle \vec{x}(0), \vec{y}(0) \rangle = \langle \vec{0}, \vec{0} \rangle$, any infinite input stream

$$s = (\vec{u}(i))_{i \in \mathbb{N}} = \vec{u}(0)\vec{u}(1)\vec{u}(2) \cdots \in (\mathbb{B}^M)^\omega$$

induces via Equations (1) and (2) an infinite sequence of consecutive states

$$\begin{aligned} c_s &= (\langle \vec{x}(i), \vec{y}(i) \rangle)_{i \in \mathbb{N}} \\ &= \langle \vec{x}(0), \vec{y}(0) \rangle \langle \vec{x}(1), \vec{y}(1) \rangle \langle \vec{x}(2), \vec{y}(2) \rangle \cdots \\ &\qquad \in ([0,1]^N \times \mathbb{B}^P)^\omega \end{aligned}$$

called the *computation* of $\mathcal{N}$ induced by the input stream s. The infinite sequence of Boolean states

$$c'_s = (\vec{y}(i))_{i \in \mathbb{N}} = \vec{y}(0)\vec{y}(1)\vec{y}(2) \cdots \in (\mathbb{B}^P)^\omega$$

is the *Boolean computation* of $\mathcal{N}$ induced by the input stream s.

Note that any B/S-RNN $\mathcal{N}$ necessarily contains finitely many possible Boolean states (indeed, if $\mathcal{N}$ possesses P Boolean output cells, then it contains at most 2^P distinct Boolean states). Consequently, for any Boolean computation c'_s, there necessarily exists at least one Boolean state that recurs infinitely often in c'_s. According to this observation, any Boolean computation c'_s consists of a finite prefix of Boolean states followed by an infinite suffix of Boolean states that repeat infinitely often – yet not necessarily in a periodic manner. The non-empty set of all the Boolean states that repeat infinitely often in c'_s will be denoted by $\infty(c'_s)$. Now, a set of

Boolean states of the form $\infty(c'_s)$ for some computation c'_s will be called an *attractor* for $\mathcal{N}$ [23]. A precise definition can be given as follows:

Definition 1. Let $\mathcal{N}$ be some B/S-RNN. A set of Boolean states $A = \{\vec{y_0}, \ldots, \vec{y_k}\} \subseteq \mathbb{B}^P$ is an *attractor* for $\mathcal{N}$ if there exists an input stream s such that the corresponding Boolean computation c'_s satisfies $\infty(c'_s) = A$.

In words, an attractor of $\mathcal{N}$ is a set of Boolean states into which the network could become forever trapped – yet not necessarily in a periodic manner –, for some Boolean computation c'_s.

In this work, we suppose that attractors can be of two distinct types, namely either *meaningful* or *spurious*. For instance, the type of each attractor could be determined by its topological features or by its neurophysiological significance with respect to measurable observations, e.g. associated with certain behaviors or sensory discriminations. For a referenced discussion about meaningful and spurious attractors in biological neural networks, see [15, Section "Neurophysiological Meaningfulness"]. The issue of the classification of the attractors into meaningful and spurious types is not the topic of this paper. Rather, from this point onwards, we assume that any B/S-RNN is already provided with a corresponding classification of all of its attractors into meaningful and spurious types.

An infinite input stream $s \in (\mathbb{B}^M)^\omega$ of $\mathcal{N}$ is then called *meaningful* if $\infty(c'_s)$ is a meaningful attractor, and it is called *spurious* if $\infty(c'_s)$ is a spurious attractor. The set of all meaningful input streams of $\mathcal{N}$ is called the *neural language* of $\mathcal{N}$ and is denoted by $L(\mathcal{N})$. An arbitrary set of input streams $L \subseteq (\mathbb{B}^M)^\omega$ is said to be *recognizable* by some B/S-RNN if there exists a network $\mathcal{N}$ such that $L(\mathcal{N}) = L$.

Finally, note that the concept of an *attractor*, when visited in a periodic way, is directly related to that of a *spatiotemporal pattern*. To illustrate this, suppose that a S/N-RNN $\mathcal{N}$ contains three Boolean output cells y_0, y_1, y_2, and that some infinite input stream s induces the corresponding Boolean computation

$$c'_s = \binom{0}{0}\binom{1}{1}\left[\binom{0}{0}\binom{1}{0}\binom{0}{1}\right]^\omega$$

For this Boolean computation c'_s, the corresponding attractor is $\infty(c'_s) = \{(0,0,1)^T, (1,0,0)^T, (0,1,1)^T\}$ and it is visited in a periodic way. The visit of this attractor by the Boolean computation c'_s corresponds precisely to the spatiotemporal pattern illustrated in Figure 2.

IV. Results

In this section, we provide a precise characterization of the computational powers of the six models of B/S-RNNs under consideration. More precisely, we first show that the neural languages recognized by B/S-RNN[$\mathbb{Q}$]s are equivalent to those recognized by deterministic Muller Turing machines, and thus belong to the $BC(\mathbf{\Pi_2^0})$ of $(\mathbb{B}^M)^\omega$ (Theorem 1). Next, we prove that the five other models of Ev$_2$-B/S-RNN[$\mathbb{Q}$]s, Ev-B/S-RNN[$\mathbb{Q}$]s, B/S-RNN[$\mathbb{R}$]s, Ev$_2$-B/S-RNN[$\mathbb{R}$]s, and Ev-B/S-RNN[$\mathbb{R}$]s are equivalent to each other, recognize the whole class of $BC(\mathbf{\Pi_2^0})$ of $(\mathbb{B}^M)^\omega$, and therefore, are all strictly more powerful than deterministic Muller Turing machines (Theorem 2). In this sense, these five neural models of computation are *super-Turing*. These results are illustrated in Figure 1 above and Table II below.

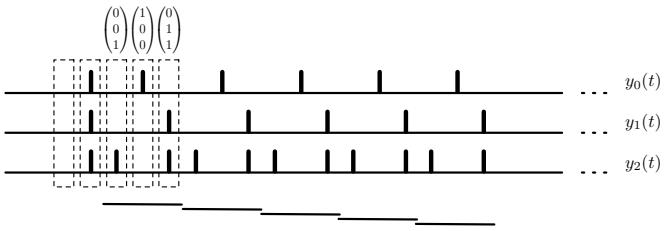

Fig. 2: The spatiotemporal pattern corresponding to periodic suffix of the Boolean computation c'_s. This spatiotemporal pattern corresponds to the periodic occurrences of the Boolean states of the attractor $\infty(c'_s) = \{(0,0,1)^T, (1,0,0)^T, (0,1,1)^T\}$. The lines under the spike raster plots indicate the successive occurrences of the spatiotemporal pattern.

TABLE II: Computational power of the six models of B/S-RNNs.

	Static	Bi-valued Evolving	Evolving
$\mathbb{Q}$	B/S-RNN[$\mathbb{Q}$]s Turing (Muller) $\in BC(\mathbf{\Pi_2^0})$	Ev$_2$-B/S-RNN[$\mathbb{Q}$]s super-Turing $= BC(\mathbf{\Pi_2^0})$	Ev-B/S-RNN[$\mathbb{Q}$]s super-Turing $= BC(\mathbf{\Pi_2^0})$
$\mathbb{R}$	B/S-RNN[$\mathbb{R}$]s super-Turing $= BC(\mathbf{\Pi_2^0})$	Ev$_2$-B/S-RNN[$\mathbb{R}$]s super-Turing $= BC(\mathbf{\Pi_2^0})$	Ev-B/S-RNN[$\mathbb{R}$]s super-Turing $= BC(\mathbf{\Pi_2^0})$

We first characterize the computational power of (static) B/S-RNN[$\mathbb{Q}$]s.

Theorem 1. *Let $L \subseteq (\mathbb{B}^M)^\omega$ be some language. Then L is recognizable by some B/S-RNN[$\mathbb{Q}$] if and only if L is recognizable by some Muller deterministic TM. In particular, if L is recognizable by some B/S-RNN[$\mathbb{Q}$], then $L \in BC(\mathbf{\Pi_2^0})$.*

Proof: Let $\mathcal{N}$ be some B/S-RNN[$\mathbb{Q}$] recognizing the language $L(\mathcal{N})$. Since the synaptic weights of $\mathcal{N}$ are rational and remain constant over time, Equations (1) and (2) are recursive, and hence, the function $f_\mathcal{N}$ described in Point (3) is also clearly recursive. Consequently, there exists some TM $\mathcal{M}$ with $N + P$ work tapes which can simulate the behavior of $\mathcal{N}$ by writing on its tapes the successive rational and Boolean activations values of the N and P internal and output cells of $\mathcal{N}$, respectively. We next provide $\mathcal{M}$ with 2^P additional designated states $q_1, \ldots, q_{2^P}$, and we modify its program in such a way that, after each simulation step, $\mathcal{M}$ enters state q_i iff $\mathcal{N}$ is in the i-th Boolean state $\vec{b}_i \in \mathbb{B}^P$, according to the lexicographic order. In this way, each infinite input stream $s \in (\mathbb{B}^M)^\omega$ induces on the one side, in the network $\mathcal{N}$, a computation c_s with an associated attractor $\infty(c'_s) \subseteq \mathbb{B}^P$, and on the other side, in the machine $\mathcal{M}$, an infinite run r_s with an associated set of sates that are visited infinitely often $\infty(r_s)$ of the form $\infty(r_s) = Q \cup Q'$,

with $Q' \subseteq \{q_1, \ldots, q_{2^P}\}$ and $Q' \neq \emptyset$. By construction, for any infinite input streams $s, s' \in (\mathbb{B}^M)^\omega$, the relation $\infty(c'_s) \neq \infty(c'_{s'}) \Rightarrow \infty(r_s) \neq \infty(r_{s'})$ holds. We can thus define the following Muller table of $\mathcal{M}$, namely $\mathcal{T} = \{\infty(r_s) : \infty(c'_s)$ is a meaningful attractor for $\mathcal{N}$, for any $s \in (\mathbb{B}^M)^\omega\}$.[2] According to this construction, one has $s \in L(\mathcal{N})$ iff $\infty(c'_s)$ is a meaningful attractor iff $\infty(r_s) \in \mathcal{T}$ iff $s \in L(\mathcal{M})$. Therefore, $L(\mathcal{N}) = L(\mathcal{M})$, showing that $L(\mathcal{N})$ is recognized by the deterministic Muller TM $\mathcal{M}$.

Conversely, let $\mathcal{M}$ be some deterministic Muller TM with table $\mathcal{T} = \{T_1, \ldots, T_k\}$ and with associated language $L(\mathcal{M})$. By the construction given in [6], there exists some sigmoid rational-weighted RNN $\mathcal{N}$ which simulates the behavior of $\mathcal{M}$. More precisely, if $\mathcal{M}$ contains n states $q_1, \ldots, q_n$, we provide $\mathcal{N}$ with P additional Boolean output cells $y_1, \ldots, y_P$, with P satisfying $2^P \geq n$, and we update the simulation process such that, during the processing of the input stream, the machine $\mathcal{M}$ visits the state q_k iff the network $\mathcal{N}$ activates the k-th Boolean state $\vec{b}_k$, according to the lexicographic order, for $k = 1, \ldots, n$. Next, for each element $T_i = \{q_{i_1}, \ldots, q_{i_{k(i)}}\}$ of the Muller table $\mathcal{T}$ of $\mathcal{M}$, we set the meaningful attractor $A_i = \{\vec{b}_{i_1}, \ldots, \vec{b}_{i_{k(i)}}\}$ in the network $\mathcal{N}$. All other possible attractors of $\mathcal{N}$ are considered to be spurious. In this way, for any infinite input stream $s \in (\mathbb{B}^M)^\omega$, the infinite run r_s of $\mathcal{M}$ satisfies $\infty(r_s) \in \mathcal{T}$ iff the Boolean computation c'_s of $\mathcal{N}$ satisfies that $\infty(c'_s)$ is a meaningful attractor. In other words, $s \in L(\mathcal{M})$ iff $s \in L(\mathcal{N})$. Therefore, $L(\mathcal{M}) = L(\mathcal{N})$, showing that $L(\mathcal{M})$ is recognized by the B/S-RNN[$\mathbb{Q}$] $\mathcal{N}$.

Finally, the second part of the Theorem comes from the previous equivalence and the fact that any language recognized by some deterministic Muller TM is in $BC(\mathbf{\Pi_2^0})$ [22]. □

We now characterize the computational powers of the five remaining models of B/S-RNNs.

Theorem 2. *The five models of Ev$_2$-B/S-RNN[$\mathbb{Q}$]s, Ev-B/S-RNN[$\mathbb{Q}$]s, B/S-RNN[$\mathbb{R}$]s, Ev$_2$-B/S-RNN[$\mathbb{R}$]s, and Ev-B/S-RNN[$\mathbb{R}$]s are all super-Turing equivalent. More precisely, for any language $L \subseteq (\mathbb{B}^M)^\omega$, the following conditions are equivalent:*

1) $L \in BC(\mathbf{\Pi_2^0})$
2) *L is recognizable by some Ev$_2$-B/S-RNN[$\mathbb{Q}$]*
3) *L is recognizable by some Ev-B/S-RNN[$\mathbb{Q}$]*
4) *L is recognizable by some B/S-RNN[$\mathbb{R}$]*
5) *L is recognizable by some Ev$_2$-B/S-RNN[$\mathbb{R}$]*
6) *L is recognizable by some Ev-B/S-RNN[$\mathbb{R}$]*

The proof of Theorem 2 is achieved via the two following Lemmas 1 and 2.

Lemma 1. *Let $L \subseteq (\mathbb{B}^M)^\omega$. If $L \in BC(\mathbf{\Pi_2^0})$, then L is recognizable by some B/S-RNN[$\mathbb{R}$] and by some Ev$_2$-B/S-RNN[$\mathbb{Q}$].*

Proof: This proof is a generalization of the one given in [14, Proposition 2]. We first consider the case of a B/S-RNN[$\mathbb{R}$], and then that of an Ev$_2$-B/S-RNN[$\mathbb{Q}$].

[2]Note that the relation $\infty(c'_s) \neq \infty(c'_{s'}) \Rightarrow \infty(r_s) \neq \infty(r_{s'})$ ensures that the table $\mathcal{T}$ is well defined, since it is impossible to have a situation where $\infty(c'_s)$ is meaningful, $\infty(c'_{s'})$ is spurious, and $\infty(r_s) = \infty(r_{s'})$, which would mean that $\infty(r_s) \in \mathcal{T}$, $\infty(r_{s'}) \notin \mathcal{T}$.

First of all, let $L \subseteq (\mathbb{B}^M)^\omega$ such that $L \in \mathbf{\Pi_2^0}$. We will consider the case of $L \in BC(\mathbf{\Pi_2^0})$ afterwards. Then L can be written as

$$L = \bigcap_{i \geq 0} \bigcup_{j \geq 0} p_{i,j} \cdot (\mathbb{B}^M)^\omega$$

where each $p_{i,j} \in (\mathbb{B}^M)^*$. Hence, a given infinite input $s \in (\mathbb{B}^M)^\omega$ belongs to L iff for all index $i \geq 0$ there exists an index $j \geq 0$ such that $s \in p_{i,j} \cdot (\mathbb{B}^M)^\omega$, or equivalently, iff for all $i \geq 0$ there exists $j \geq 0$ such that $p_{i,j} \subsetneq s$. Besides, as described in details in [14], one can show that the infinite sequence $(p_{i,j})_{i,j \in \mathbb{N}}$ can be encoded into some single real number r_L such that, for any pair of indices $(i, j) \in \mathbb{N} \times \mathbb{N}$, the decoding procedure of $(r_L, i, j) \mapsto p_{i,j}$ is actually recursive.

According to these considerations, the problem of determining whether some input $s \in (\mathbb{B}^M)^\omega$ provided step by step belongs to L or not can be decided in infinite time by the Algorithm 1 given below. This algorithm consists of two subroutines performed in parallel. It uses the designated real number r_L (on line 12), and it is designed in such a precise way that, on every input $s \in (\mathbb{B}^M)^\omega$, it returns infinitely many 1's iff $s \in \bigcap_{i \geq 0} \bigcup_{j > 0} p_{i,j} \cdot (\mathbb{B}^M)^\omega = L$. Moreover, note that if the designated real number r_L is provided in advance, then every step of Algorithm 1 is actually recursive.

Consequently, according to the real time computational equivalence between rational-weighted RNNs and TMs [7], there exists some RNN[$\mathbb{Q}$] $\mathcal{N}_1$ such that, if the real number r_L is given in advance as the activation value of one of its neuron x, then $\mathcal{N}_1$ is actually capable of simulating the behavior of Algorithm 1. In particular, to perform line 4, one uses M distinct cells in order to store the M Boolean components of $\vec{s}(t)$ (see [7] for further details). Consequently, if one adds to x a background synaptic connection of real intensity r_L, one obtains a RNN[$\mathbb{R}$] $\mathcal{N}_2$ capable of simulating Algorithm 1. Hence, if one further adds to $\mathcal{N}_2$ an additional binary output cell y which is designed to take value 1 every time Algorithm 1 returns a 1, one obtains a B/S-RNN[$\mathbb{R}$] $\mathcal{N}$ such that, on every input $s \in (\mathbb{B}^M)^\omega$, the only output cell y will produce infinitely many 1's iff Algorithm 1 will return infinitely many 1's, namely iff $s \in \bigcap_{i \geq 0} \bigcup_{j \geq 0} p_{i,j} \cdot (\mathbb{B}^M)^\omega = L$. Consequently, by taking $\{(1)\}$ and $\{(0), (1)\}$ as the two sole meaningful attractors of $\mathcal{N}$, one has $L(\mathcal{N}) = L$, meaning that L is recognized by some B/S-RNN[$\mathbb{R}$].

We now modify the proof in order to capture the case of an Ev$_2$-B/S-RNN[$\mathbb{Q}$]. Towards, this purpose, we first mention that the infinite sequence $(p_{i,j})_{i,j \in \mathbb{N}}$ can be encoded into some infinite word $w_L \in \{0, 1\}^\omega$ such that, for any pair of indices $(i, j) \in \mathbb{N} \times \mathbb{N}$, the decoding procedure of $(w_L, i, j) \mapsto p_{i,j}$ is actually recursive. According to these considerations, we modify Algorithm 1 by assuming that it receives the designated infinite word w_L bit by bit instead of having the designated real number r_L be provided in advance. One then replaces lines 11 and 12 by the following two ones:

11: wait until $p_{i,j}$ has been encoded in w_L and until $c \geq |p_{i,j}|$
12: decode $p_{i,j}$ from w_L

Algorithm 1 can be performed by some Ev$_2$-B/S-RNN[$\mathbb{Q}$]. Indeed, in the B/S-RNN[$\mathbb{R}$] $\mathcal{N}$ described above, one replaces the static background activity of neuron x of real intensity

Algorithm 1 Procedure which uses the designated real number r_L

Require: Input $s = \vec{s}(0)\vec{s}(1)\vec{s}(2)\cdots \in (\mathbb{B}^M)^\omega$ provided step by step at successive time steps $t = 0, 1, 2, \ldots$

1: **SUBROUTINE 1**
2: $c \leftarrow 0$ // c counts the number of letters of s
3: **for all** time step $t \geq 0$ **do**
4: store each incoming Boolean vector $\vec{s}(t) \in \mathbb{B}^M$
5: $c \leftarrow c + 1$
6: **end for**
7: **END SUBROUTINE 1**

8: **SUBROUTINE 2**
9: $i \leftarrow 0$, $j \leftarrow 0$
10: **loop**
11: wait until $c \geq |p_{i,j}|$ // wait until input s becomes at least as long as $p_{i,j}$
12: decode $p_{i,j}$ from r_L // recursive procedure if r_L is given in advance, cf. [14]
13: **if** $p_{i,j} \subseteq s[0{:}c]$ **then** // in this case, $s \in p_{i,j} \cdot (\mathbb{B}^M)^\omega$
14: **return** 1 // for this i, there exists a j such that $s \in p_{i,j} \cdot (\mathbb{B}^M)^\omega$
15: $i \leftarrow i + 1$, $j \leftarrow 0$ // begin to test if $s \in p_{i+1,0} \cdot (\mathbb{B}^M)^\omega$
16: **else** // in this case, $s \notin p_{i,j} \cdot (\mathbb{B}^M)^\omega$
17: **return** 0 // for this i, there is up to now no j such that $s \in p_{i,j} \cdot (\mathbb{B}^M)^\omega$
18: $i \leftarrow i$, $j \leftarrow j + 1$ // begin to test if $s \in p_{i,j+1} \cdot (\mathbb{B}^M)^\omega$
19: **end if**
20: **end loop**
21: **END SUBROUTINE 2**

r_L by an evolving background activity of intensities $w_L = w_L(0)w_L(1)w_L(2)\cdots \in \{0,1\}^\omega$. In this way, one obtains a network $\mathcal{N}'$ whose all static synaptic weights are rational and whose only evolving synaptic weight is bi-valued. We next slightly modify this networks in order to perform correctly the recursive updated lines 11 and 12 of Algorithm 1. One obtains an Ev$_2$-B/S-RNN[$\mathbb{Q}$] $\mathcal{N}'$ such that, on every input $s \in (\mathbb{B}^M)^\omega$, the only output cell y of $\mathcal{N}'$ will produce infinitely many 1's iff updated Algorithm 1 will return infinitely many 1's, namely iff $s \in \bigcap_{i\geq 0} \bigcup_{j\geq 0} p_{i,j} \cdot (\mathbb{B}^M)^\omega = L$. Therefore, by taking $\{(1)\}$ and $\{(0), (1)\}$ as the two sole meaningful attractors of $\mathcal{N}'$, one obtains $L(\mathcal{N}') = L$, meaning that L is recognized by some Ev$_2$-B/S-RNN[$\mathbb{Q}$].

This concludes the proof for the case of $L \in \mathbf{\Pi_2^0}$. We finally extend the proof for the case of $L \in BC(\mathbf{\Pi_2^0})$. Towards this purpose, we show that any finite union and complementation of a $\mathbf{\Pi_2^0}$ set can also be recognized by some B/S-RNN[$\mathbb{R}$] and by some Ev$_2$-B/S-RNN[$\mathbb{Q}$].

Firstly, let $L = L_1 \cup L_2$ such that $L_i \in \mathbf{\Pi_2^0}$, for $i = 1, 2$. By the previous arguments, there exist two B/S-RNN[$\mathbb{R}$]s (or two Ev$_2$-B/S-RNN[$\mathbb{Q}$]s) $\mathcal{N}_1$ and $\mathcal{N}_2$ which recognize L_1 and L_2, respectively. By suitably merging $\mathcal{N}_1$ and $\mathcal{N}_2$ into some new network $\mathcal{N}$, and by setting as meaningful attractors of $\mathcal{N}$ all those involving at least one Boolean state for which at least one of the two output cells is spiking, one obtains a B/S-RNN[$\mathbb{R}$] (or an Ev$_2$-B/S-RNN[$\mathbb{Q}$]) $\mathcal{N}$ that recognizes $L_1 \cup L_2$. In other words, one has $L(\mathcal{N}) = L_1 \cup L_2 = L$.

Secondly, let $L \in \mathbf{\Sigma_2^0}$. Then by definition, $L^\complement \in \mathbf{\Pi_2^0}$. By the previous arguments, there exists a B/S-RNN[$\mathbb{R}$] (or an Ev$_2$-B/S-RNN[$\mathbb{Q}$]) $\mathcal{N}$ which recognizes $L^\complement$ via some relevant simulation of Algorithm 1. We now update $\mathcal{N}$ in a way that its only binary output cell y takes value 1 every time Algorithm 1 returns a 0 (instead of 1). One thus obtains a B/S-RNN[$\mathbb{R}$] (or an Ev$_2$-B/S-RNN[$\mathbb{Q}$]) $\mathcal{N}'$ such that, on every input $s \in (\mathbb{B}^M)^\omega$, the only output cell y of $\mathcal{N}'$ will produce infinitely many 1's as well as only finitely many 0's iff Algorithm 1 will return infinitely many 0's and finitely many 1's, i.e. iff $s \notin L^\complement$, i.e. iff $s \in L$. Consequently, by taking $\{(1)\}$ as the sole meaningful attractor of $\mathcal{N}'$, one obtains a B/S-RNN[$\mathbb{R}$] (or an Ev$_2$-B/S-RNN[$\mathbb{Q}$]) $\mathcal{N}'$ that recognizes L. In other terms, $L(\mathcal{N}') = L$.

Consequently, any finite union and complementation of a $\mathbf{\Pi_2^0}$ set can be recognized by some B/S-RNN[$\mathbb{R}$] and by some Ev$_2$-B/S-RNN[$\mathbb{Q}$]. In other words, if $L \in BC(\mathbf{\Pi_2^0})$, then L is recognizable by some B/S-RNN[$\mathbb{R}$] and by some Ev$_2$-B/S-RNN[$\mathbb{Q}$]. □

Lemma 2. *Let $L \subseteq (\mathbb{B}^M)^\omega$. If is recognizable by some Ev-B/S-RNN[$\mathbb{R}$], then $L \in BC(\mathbf{\Pi_2^0})$.*

Proof: Let $L \subseteq (\mathbb{B}^M)^\omega$ be recognizable by some Ev-B/S-RNN[$\mathbb{R}$] $\mathcal{N}$. Suppose that $\mathcal{N}$ contains the K meaningful attractors $A_i = \{\vec{b}_{i_1}, \ldots, \vec{b}_{i_{k(i)}}\}$, for $i = 1, \ldots, K$, where $1 \leq i_1 < \ldots < i_{k(i)} \leq 2^P$, and where $\vec{b}_n$ denotes the n-th Boolean vector of $\mathbb{B}^P$ according to the lexicographic order.

Note that the dynamics of $\mathcal{N}$ can naturally be associated with the function $g_\mathcal{N} : (\mathbb{B}^M)^\omega \to (\mathbb{B}^P)^\omega$ defined by $g_\mathcal{N}(s) = c'_s$, where c'_s is the Boolean computation generated by $\mathcal{N}$ when the infinite input stream s is received. The nature of our dynamics ensures that this function is sequential, i.e., for any time step $t \geq 0$, the Boolean vectors $\vec{s}(t)$ and $\vec{c'_s}(t)$ are generated simultaneously. Hence, $g_\mathcal{N}$ is Lipschitz and thus continuous, cf. [14, Lemma 1]. Besides, we recall for the sequel that the preimage of a $\mathbf{\Pi_2^0}$ or a $\mathbf{\Sigma_2^0}$ set by some continuous functions is also a $\mathbf{\Pi_2^0}$ or a $\mathbf{\Sigma_2^0}$ set, respectively.

According to these considerations, the language $L(\mathcal{N})$ can

be expressed by the following sequence of equations:

$$
\begin{aligned}
L(\mathcal{N}) &= \Big\{ s \in (\mathbb{B}^M)^\omega : \infty(c'_s) \text{ is a meaningful attractor} \Big\} \\
&= \Big\{ s \in (\mathbb{B}^M)^\omega : \infty(c'_s) = A_i \text{ for some } i = 1, \ldots, K \Big\} \\
&= \bigcup_{i=1}^{K} \Big\{ s \in (\mathbb{B}^M)^\omega : \infty(c'_s) = A_i \Big\} \\
&= \bigcup_{i=1}^{K} \Big\{ s \in (\mathbb{B}^M)^\omega : \text{for all } j \in \{i_1, \ldots, i_{k(i)}\}, \\
&\qquad\qquad g_{\mathcal{N}}(s) \text{ contains infinitely many } \vec{b}_j\text{'s, and} \\
&\qquad\qquad \text{for all } j \in \{1, \ldots, 2^P\} \setminus \{i_1, \ldots, i_{k(i)}\}, \\
&\qquad\qquad g_{\mathcal{N}}(s) \text{ contains finitely many } \vec{b}_j\text{'s} \Big\} \\
&= \bigcup_{i=1}^{K} \Big[\bigcap_{j \in \{i_1, \ldots, i_{k(i)}\}} \Big\{ s \in (\mathbb{B}^M)^\omega : g_{\mathcal{N}}(s) \text{ has infinitely many } \vec{b}_j\text{'s} \Big\} \cap \\
&\qquad \bigcap_{j \in \{1, \ldots, 2^P\} \setminus \{i_1, \ldots, i_{k(i)}\}} \Big\{ s \in (\mathbb{B}^M)^\omega : g_{\mathcal{N}}(s) \text{ has finitely many } \vec{b}_j\text{'s} \Big\} \Big] \\
&= \bigcup_{i=1}^{K} \Big[\bigcap_{j \in \{i_1, \ldots, i_{k(i)}\}} \Big\{ s \in (\mathbb{B}^M)^\omega : \\
&\qquad\qquad g_{\mathcal{N}}(s) \in \underbrace{\bigcap_{n \geq 0} \bigcup_{m \geq 0} (\mathbb{B}^P)^{n+m} \cdot \vec{b}_j \cdot (\mathbb{B}^P)^\omega}_{\substack{c'_s \text{ contains infinitely many } \vec{b}_j\text{'s, i.e.} \\ \forall n \geq 0 \; \exists m \geq n \; c'_s(n+m) = \vec{b}_j \\ \text{thus in } \mathbf{\Pi_2^0}}} \Big\} \cap \\
&\qquad \bigcap_{j \in \{1, \ldots, 2^P\} \setminus \{i_1, \ldots, i_{k(i)}\}} \Big\{ s \in (\mathbb{B}^M)^\omega : \\
&\qquad\qquad g_{\mathcal{N}}(s) \in \underbrace{\Big(\bigcap_{n \geq 0} \bigcup_{m \geq 0} (\mathbb{B}^P)^{n+m} \cdot \vec{b}_j \cdot (\mathbb{B}^P)^\omega \Big)^{\complement}}_{\substack{c'_s \text{ contains only finitely many } \vec{b}_j\text{'s, i.e.} \\ \text{complement of a } \mathbf{\Pi_2^0}\text{-set} \\ \text{thus in } \mathbf{\Sigma_2^0}}} \Big\} \Big] \\
&= \bigcup_{i=1}^{K} \Big[\bigcap_{j \in \{i_1, \ldots, i_{k(i)}\}} \underbrace{g_{\mathcal{N}}^{-1}\Big(\bigcap_{n \geq 0} \bigcup_{m \geq 0} (\mathbb{B}^P)^{n+m} \cdot \vec{b}_j \cdot (\mathbb{B}^P)^\omega \Big)}_{\text{preimage by a continuous function of a } \mathbf{\Pi_2^0}\text{-set, thus in } \mathbf{\Pi_2^0}} \cap \\
&\qquad \bigcap_{j \in \{1, \ldots, 2^P\} \setminus \{i_1, \ldots, i_{k(i)}\}} \underbrace{g_{\mathcal{N}}^{-1}\Big(\Big(\bigcap_{n \geq 0} \bigcup_{m \geq 0} (\mathbb{B}^P)^{n+m} \cdot \vec{b}_j \cdot (\mathbb{B}^P)^\omega \Big)^{\complement} \Big)}_{\text{preimage by a continuous function of a } \mathbf{\Sigma_2^0}\text{-set, thus in } \mathbf{\Sigma_2^0}} \Big]
\end{aligned}
$$

It follows that $L(\mathcal{N}) \in BC(\mathbf{\Pi_2^0})$, since it consists of finite unions and finite intersections of $\mathbf{\Pi_2^0}$ and $\mathbf{\Sigma_2^0}$ sets. □

Proof of Theorem 2: Let $L \subseteq (\mathbb{B}^M)^\omega$ such that $L \in BC(\mathbf{\Pi_2^0})$. By Lemma 1, L is recognized by some Ev_2-B/S-RNN[$\mathbb{Q}$] and by some B/S-RNN[$\mathbb{R}$]. According to the relationships between the computational powers of B/S-RNNs described in Figure 1, L is also recognizable by some Ev-B/S-RNN[$\mathbb{Q}$], Ev_2-B/S-RNN[$\mathbb{R}$], and Ev-B/S-RNN[$\mathbb{R}$]. This proves the five implications from Point (1) to Points (2), (3), (4), (5), and (6).

Conversely, let L be recognized by some Ev_2-B/S-RNN[$\mathbb{Q}$], some Ev-B/S-RNN[$\mathbb{Q}$], some B/S-RNN[$\mathbb{R}$], or some Ev_2-B/S-RNN[$\mathbb{R}$]. By the relationships between the computational powers of B/S-RNNs described in Figure 1, L is also recognizable by some Ev-B/S-RNN[$\mathbb{R}$]. By Lemma 2, $L \in BC(\mathbf{\Pi_2^0})$. This proves the five other implications from Points (2), (3), (4), (5), and (6) to Point (1). □

V. Discussion

We have introduced the concept of a *hybrid recurrent neural network* – or *Boolean/sigmoid recurrent neural network* (B/S-RNN) – composed with Boolean input and output cells as well as sigmoid internal cells. When subjected to some infinite binary input stream, the Boolean output cells necessarily exhibit some attractor dynamics, which is assumed to be of two possible kinds, i.e. either meaningful or spurious, determined for instance by some neurophysiological significance with respect to measurable observations. The attractor dynamics are the precise phenomena that underly the arising of spatiotemporal patterns of discharges elicited by the output Boolean cells. We have characterized the computational power of the hybrid neural networks in terms of their attractor dynamics.

More precisely, we have considered six different models of B/S-RNNs according to whether their synaptic weights are modelled by rational or real numbers, and to whether the these synaptic weights are either of a static nature, or able to evolve over time among only two possible values, or able to evolve over time among any possible values between two designated bounds. We have characterized the computational power of these recurrent neural networks as the topological complexity of the set of input streams – referred to as their neural language – which lead them to the visit of some meaningful attractors. In this context, we have proven that rational-weighted static B/S-RNNs are equivalent to deterministic Muller Turing machines (Theorem 1), and that the five other models of B/S-RNNs are, on the one hand, computationally equivalent one to the other, and, on the other hand, strictly more powerful than deterministic Muller Turing machines, hence capable of a *super-Turing* computational power (Theorem 2). Precise mathematical characterizations of the neural languages recognized by those B/S-RNNs are further provided. These results are summarized in Figure 1 and Table II. They extend those undertaken in [14].

Our achievements show that the incorporation of only bi-valued evolving capabilities into rational recurrent neural networks suffices to break the Turing barrier and achieve the super-Turing level of computation. The consideration of any more general mechanisms of architectural evolvability or the incorporation of real synaptic weights in the model would actually not further increase the capabilities of the neural networks. Consequently, these considerations allow us to drop any kind of analogue or complex evolving-based assumption, and keep only bi-valued synaptic plasticity in the neural models under consideration, in order to achieve a maximal computational power. These results provide a precise generalization to the current computational context of those obtained for the cases of classical as well as interactive computations [10], [11], [16], [18]–[20]. They further support the idea that recurrent neural networks provide a natural model of computation beyond the Turing limits [17].

Our attractor-based approach to the computational capabilities of recurrent neural networks is justified by the fact that, in our model, the attractor dynamics of the neural networks are the precise phenomena that underly the arising of spatiotemporal patterns of discharges – a feature that we consider to be significantly involved in the processing of information in the brain. In fact, several evidence exist of spatiotemporal firing patterns in behaving animals, from rats to primates [24], [25], where preferred firing sequences can be associated to specific types of stimuli and behaviours. Such patterns have also been observed in large scale neural network simulations [26] as the

outcome of dynamical changes in synaptic weights, e.g. see Figure 3. Besides, periodic attractors that are simple cases of limit cycles, as well as attractors with non-integer, fractal dimensions have been observed in experimental neuronal data [27], [28], as well as in neuronal network simulations [29]. Hence, despite being totally associated to deterministic dynamics, the nonlinearities and the sensitivity to the initial conditions make the trajectories of these fractal attractors totally chaotic and let emerge strange attractors.

For future work, practical studies concerning the computational capabilities of biological or artificial neural networks based on their attractor dynamics are envisioned to be carried on in light of the present theoretical results. Furthermore, the study of the computational capabilities of more biologically-oriented neural models involved in more bio-inspired computational frameworks is expected to be pursued.

From a general perspective, we believe that such comparative studies between the computational capabilities of neural models and abstract machines might bring further insight to the understanding of the intrinsic natures of both biological as well as artificial intelligences.

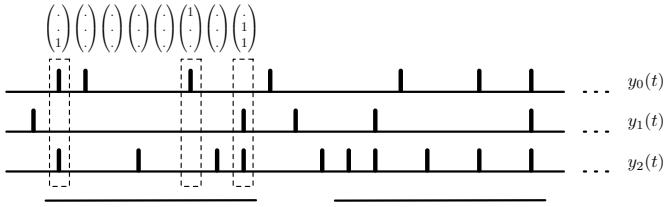

Fig. 3: The spatiotemporal pattern corresponding to a recurrent sequence of the Boolean computation c'_s. In this case the recurrent occurrences of each state of the attractor $\infty(c'_s) = \{(.,.,1)^T, (1,.,.)^T, (.,1,1)^T\}$ are defined following precise time intervals: $\langle y_0, y_1, y_1, y_2; 5, 2, 0\rangle$.This pattern corresponds to a spike of cell y_0 followed 5 time units later by a spike of cell y_1, then another spike of cell y_1 after 2 more time units (i.e. t time units from pattern onset), and a spike of cell y_2 after 0 t.u. (that means at the same time) from the previous event.

REFERENCES

[1] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysic*, vol. 5, pp. 115–133, 1943.

[2] A. M. Turing, "Intelligent machinery," National Physical Laboratory, Teddington, UK, Technical Report, 1948.

[3] S. C. Kleene, "Representation of events in nerve nets and finite automata," in *Automata Studies*, C. Shannon and J. McCarthy, Eds. Princeton, NJ: Princeton University Press, 1956, pp. 3–41.

[4] J. v. Neumann, *The computer and the brain*. New Haven, CT, USA: Yale University Press, 1958.

[5] M. L. Minsky, *Computation: finite and infinite machines*. Englewood Cliffs, N. J.: Prentice-Hall, Inc., 1967.

[6] H. T. Siegelmann and E. D. Sontag, "Analog computation via neural networks," *Theor. Comput. Sci.*, vol. 131, no. 2, pp. 331–360, 1994.

[7] ——, "On the computational power of neural nets," *J. Comput. Syst. Sci.*, vol. 50, no. 1, pp. 132–150, 1995.

[8] H. T. Siegelmann, *Neural networks and analog computation: beyond the Turing limit*. Cambridge, MA, USA: Birkhauser Boston Inc., 1999.

[9] J. Síma and P. Orponen, "General-purpose computation with neural networks: A survey of complexity theoretic results," *Neural Computation*, vol. 15, no. 12, pp. 2727–2778, 2003.

[10] J. Cabessa and H. T. Siegelmann, "Evolving recurrent neural networks are super-turing," in *The 2011 International Joint Conference on Neural Networks, IJCNN 2011, San Jose, California, USA, July 31 - August 5, 2011*. IEEE, 2011, pp. 3200–3206.

[11] ——, "The super-Turing computational power of plastic recurrent neural networks," *Int. J. Neural Syst.*, vol. 24, no. 8, 2014.

[12] J. Cabessa and A. E. P. Villa, "A hierarchical classification of first-order recurrent neural networks," in *Language and Automata Theory and Applications, 4th International Conference, LATA 2010, Trier, Germany, May 24-28, 2010. Proceedings*, ser. Lecture Notes in Computer Science, A. H. D. et al., Ed., vol. 6031. Springer, 2010, pp. 142–153.

[13] ——, "A Hierarchical Classification of First-Order Recurrent Neural Networks," *Chinese Journal of Physiology*, vol. 53, pp. 407–416, 2010.

[14] ——, "The expressive power of analog recurrent neural networks on infinite input streams," *Theor. Comput. Sci.*, vol. 436, pp. 23–34, 2012.

[15] ——, "An Attractor-Based Complexity Measurement for Boolean Recurrent Neural Networks," *PLoS ONE*, vol. 9, no. 4, p. e94204, 2014.

[16] J. Cabessa and H. T. Siegelmann, "The computational power of interactive recurrent neural networks," *Neural Computation*, vol. 24, no. 4, pp. 996–1019, 2012.

[17] J. Cabessa and A. E. P. Villa, "Recurrent neural networks - a natural model of computation beyond the Turing limits," in *IJCCI*, A. C. R. et al., Ed. SciTePress, 2012, pp. 594–599.

[18] ——, "The super-Turing computational power of interactive evolving recurrent neural networks," in *Artificial Neural Networks and Machine Learning - ICANN 2013 - 23rd International Conference on Artificial Neural Networks, Sofia, Bulgaria, September 10-13, 2013. Proceedings*, ser. Lecture Notes in Computer Science, V. M. et al., Ed., vol. 8131. Springer, 2013, pp. 58–65.

[19] ——, "Interactive evolving recurrent neural networks are super-Turing universal," in *Artificial Neural Networks and Machine Learning - ICANN 2014 - 24th International Conference on Artificial Neural Networks, Hamburg, Germany, September 15-19, 2014. Proceedings*, ser. Lecture Notes in Computer Science, S. W. et al., Ed., vol. 8681. Springer, 2014, pp. 57–64.

[20] J. Cabessa, "Interactive evolving recurrent neural networks are super-Turing," in *ICAART (1)*, J. Filipe and A. L. N. Fred, Eds. SciTePress, 2012, pp. 328–333.

[21] J. P. Segundo, G. P. Moore, L. J. Stensaas, and T. H. Bullock, "Sensitivity of neurones in aplysia to temporal pattern of arriving impulses," *J Exp Biol*, vol. 40, pp. 643–667, Dec 1963.

[22] D. Perrin and J.-E. Pin, *Infinite Words*, ser. Pure and Applied Mathematics. Elsevier, 2004, vol. 141.

[23] S. A. Kauffman, *The origins of order: Self-organization and selection in evolution*. New York: Oxford University Press, 1993.

[24] E. Vaadia, I. Haalman, M. Abeles, H. Bergman, Y. Prut, H. Slovin, and A. Aertsen, "Dynamics of neuronal interactions in monkey cortex in relation to behavioural events," *Nature*, vol. 373, no. 6514, pp. 515–518, 1995.

[25] A. E. P. Villa, I. V. Tetko, B. Hyland, and A. Najem, "Spatiotemporal activity patterns of rat cortical neurons predict responses in a conditioned task," *Proc Natl Acad Sci U S A*, vol. 96, no. 3, pp. 1106–1111, 1999.

[26] J. Iglesias and A. E. P. Villa, "Recurrent spatiotemporal firing patterns in large spiking neural networks with ontogenetic and epigenetic processes," *J Physiol Paris*, vol. 104, no. 3-4, pp. 137–146, 2010.

[27] A. Celletti and A. E. P. Villa, "Low-dimensional chaotic attractors in the rat brain," *Biol Cybern*, vol. 74, no. 5, pp. 387–393, 1996.

[28] A. E. P. Villa, I. V. Tetko, A. Celletti, and A. Riehle, "Chaotic dynamics in the primate motor cortex depend on motor preparation in a reaction-time task," *Current Psychology of Cognition*, vol. 17, pp. 763–780, 1998.

[29] J. Iglesias, O. Chibirova, and A. Villa, "Nonlinear dynamics emerging in large scale neural networks with ontogenetic and epigenetic processes," *Lecture Notes in Computer Science*, vol. 4668, pp. 579–588, 2007.