Finite State Machines and Recurrent Neural Networks – Automata and Dynamical Systems Approaches

Peter Tiňo[a,c], Bill G. Horne [c], C. Lee Giles[c,d], Pete C. Collingwood[b]

[a]Department of Computer Science and
Engineering
Slovak Technical University
Ilkovicova 3, 812 19 Bratislava, Slovakia
Email: tino@decef.elf.stuba.sk

[b]School of Computing & Man. Sci.
Sheffield Hallam University
Hallam Business Park
100 Napier St., Sheffield, S11 8HD, UK
Email: p.c.collingwood@shu.ac.uk

[c]NEC Research Institute
4 Independence Way
Princeton, NJ 08540
Email:
{tino,horne,giles}@research.nj.nec.com

[d]Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742

Technical Report
UMIACS-TR-95-1 and CS-TR-3396
Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742

Finite State Machines and Recurrent Neural Networks – Automata and Dynamical Systems Approaches

Peter Tiňo[a,c], Bill G. Horne [c], C. Lee Giles[c,d], Pete C. Collingwood[b]

[a]Department of Computer Science and Engineering
Slovak Technical University
Ilkovicova 3, 812 19 Bratislava, Slovakia
Email: tino@decef.elf.stuba.sk

[c]NEC Research Institute
4 Independence Way
Princeton, NJ 08540
Email:
{tino,horne,giles}@research.nj.nec.com

[b]School of Computing & Man. Sci.
Sheffield Hallam University
Hallam Business Park
100 Napier St., Sheffield, S11 8HD, UK
Email: p.c.collingwood@shu.ac.uk

[d]Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742

Abstract

We present two approaches to the analysis of the relationship between a recurrent neural network (RNN) and the finite state machine $\mathcal{M}$ the network is able to exactly mimic. First, the network is treated as a state machine and the relationship between the RNN and $\mathcal{M}$ is established in the context of algebraic theory of automata. In the second approach, the RNN is viewed as a set of discrete-time dynamical systems associated with input symbols of $\mathcal{M}$. In particular, issues concerning network representation of loops and cycles in the state transition diagram of $\mathcal{M}$ are shown to provide a basis for the interpretation of learning process from the point of view of bifurcation analysis. The circumstances under which a loop corresponding to an input symbol x is represented by an attractive fixed point of the underlying dynamical system associated with x are investigated. For the case of two recurrent neurons, under some assumptions on weight values, bifurcations can be understood in the geometrical context of intersection of increasing and decreasing parts of curves defining fixed points. The most typical bifurcation responsible for the creation of a new fixed point is the saddle node bifurcation.

1 Introduction

The relationship between recurrent neural networks (RNN) and automata has been treated by many [29], [26], [8], [10], [15], [17], [5], [38], [39], [28], [11], [23]. State units' activations represent past histories and clusters of these activations can represent the states of the generating automaton [18].

In this contribution, the relationship between a RNN and a finite state machine it exactly mimics is investigated from two points of view. First (section 5), the network is treated as a state machine. The concept of state equivalence is used to reduce the infinite, non-countable set of network states (activations of RNN state neurons) to a finite factor state set corresponding to the set of states of $\mathcal{M}$. Second (section 6), the RNN is viewed as a set of discrete-time dynamical systems associated with input symbols of $\mathcal{M}$. The dynamical systems operate on $(0,1)^L$, where L is the number of recurrent neurons of the RNN. In our experiments, loops and cycles corresponding to an input symbol x of $\mathcal{M}$ have stable representation as attractive fixed points and periodic orbits respectively of the dynamical system associated with the input x. Suppose there is a loop associated with an input x in a state q of $\mathcal{M}$. Denote the set of network states equivalent to q by $(q)_{\mathcal{N}}$. Then, if there is a vertex $v \in \{0,1\}^L$ such that v is in the closure of $(q)_{\mathcal{N}}$, the loop is likely to be represented by an attractive fixed point[1] "near" v.

A related work was independently done by Casey [5], [6]. In his setting, RNN is assumed to operate in a noisy environment (representing for example a noise corresponding to round-off errors in computations performed on a digital computer). RNNs are trained to perform grammatical inference. It is proved that a presence of a loop in the state transition diagram of the automaton [2] necessarily implies the presence of an attractive set inside RNN state space (see the discussion in section 6). It is also shown that the method for extraction of an automaton from a trained RNN introduced in [17] is consistent: the method is based on dividing RNN state space into equal hypercubes and there is always a finite number of hypercubes one needs to unambiguously cover regions of equivalent network states.

In section 7 a more detailed analysis of the case when RNN has two state neurons is presented.

[1] of the corresponding dynamical system

[2] recognizing the same language as the RNN

Under some conditions on weight values, the number, position and stability types of fixed points of the underlying dynamical systems are analyzed and bifurcation mechanism is clarified. The most typical bifurcation responsible for the creation of a new fixed point as the saddle node bifurcation. A mechanism of correct behaviour of RNN for short input strings, when for long strings, the network is known to generalize poorly is investigated in section 8. In such cases, a correct state transition diagram of a FSM the network was trained with can still be extracted [17]. A tool called the state degradation diagram is developed to illustrate how regions of network state space, initially acting as if they assumed the role of states of the FSM in which there is a loop associated with an input symbol x, gradually degradate upon repeated presentation of x. Sections 2 and 3 bring brief introductions to state machines and dynamical systems respectively. Section 4 is devoted to the model of RNN [31] used for learning FSMs.

2 State Machines

This section introduces the concept of state machine, which is a generalized finite state machine with possibly uncountable number of states. When viewed as automata, RNNs can be described in terms of state machines.

A *state machine* (SM) is a 6-tuple $\mathcal{M}=(X,Y,S,f_s,f_o,s_0)$, where

- X is a nonempty finite set called the input set
- Y is a nonempty finite set called the output set
- S is a nonempty set called the set of internal states
- f_s is a map $f_s : S \times X \rightarrow S$ called the next-state function
- f_o is a map $f_o : S \times X \rightarrow Y$ called the output function
- $s_0 \in S$ is called the *initial state*

SMs with a finite internal state set are called *finite state machines* (FSMs).

We assume that the reader is familiar with the notion of monoid of words over a finite set. Following the standard notation, Λ, X^*, X^+ and uv denote the empty word, the set of all words over X, the set of all nonempty words over X, and concatenation of words u and v respectively.

In every moment $\mathcal{M}$ is in exactly one state $s \in S$. When an element $x \in X$ is read in, the machine changes its state to $f_s(s,x)$ and yields the output $f_o(s,x)$. The processing of any input word $w \in X^+$ by $\mathcal{M}$ always starts with $\mathcal{M}$ being in the initial state.

If for some $x \in X$ and $s \in S$, it holds $f_s(s,x) = s$, then it is said that there is an *x-loop in the state s*. If there exist m $(m \geq 2)$ distinct states $s_1, ..., s_m \in S$ and an input $x \in X$, such that $f_s(s_i, x) = s_{i+1}$, for all $i = 1, ..., m-1$ and $f_s(s_m, x) = s_1$, then the set $\{s_1, ..., s_m\}$ is said to be an *x-cycle of length m passing through the states $s_1, ..., s_m$*.

It is convenient to extend the domain of f_s and f_o from $S \times X$ to $S \times X^*$ and $S \times X^+$ respectively:

- $\forall s \in S;\ \ f_s(s, \Lambda) = s,$
- $\forall s \in S,\ \forall w \in X^*,\ \forall x \in X;\ \ f_s(s, wx) = f_s(f_s(s,w), x)$ and $f_o(s, wx) = f_o(f_s(s,w), x)$.

Yet further generalization of f_o is useful:

$$\forall s \in S,\ \forall w = x_1 x_2 ... x_n \in X^+;\ \ f_o^+(s,w) = f_o(s, x_1) f_o(s, x_1 x_2) ... f_o(s, x_1 x_2 ... x_n).$$

A *distinguishing sequence of $\mathcal{M}$* is a word $w \in X^+$ such that there are no two states s_1, s_2 of $\mathcal{M}$ for which $f_o^+(s_1, w) = f_o^+(s_2, w)$.

The *behaviour of $\mathcal{M}$* is a map $B_{\mathcal{M}} : X^+ \to Y$: $\forall w \in X^+;\ \ B_{\mathcal{M}}(w) = f_o(s_0, w)$.

A *state* $s_2 \in S$ is said to be *accessible* and *x-accessible from the state* $s_1 \in S$ if there exists some $w \in X^*$ and $w \in \{x\}^*$ respectively, such that $s_2 = f_s(s_1, w)$. M is said to be *connected* if every state $s \in S$ is accessible from s_0. The set of all states that are x-accessible from a state $s \in S$ is denoted by $Acc(x,s)$. An x-cycle $\gamma = \{s_1, ..., s_m\}$ is said to be x-accessible from a state $p \in S$, if $\gamma \subseteq Acc(x,p)$.

An input *word* $w \in X^*$ is *leading to a state* q if $f_s(s_0, w) = q$. An input word leading to q is minimal if there is no input word leading to q of shorter length.

We shall also need some concepts concerning state and machine equivalence. Let $\mathcal{M}_i = (X, Y, S_i, f_s^i, f_o^i, s_{0i}), i = 1, 2$ be two SMs. *States* $s_1 \in S_1$ and $s_2 \in S_2$ are said to be *equivalent* if there is no non-empty word over X which would cause $\mathcal{M}_1$ to give different output from that given by $\mathcal{M}_2$, provided $\mathcal{M}_1$ and $\mathcal{M}_2$ started from s_1 and s_2 respectively. This is formally represented by the equivalence relation $E(\mathcal{M}_1, \mathcal{M}_2) \subseteq S_1 \times S_2$:

$$(s_1, s_2) \in E(\mathcal{M}_1, \mathcal{M}_2) \ \ \textit{iff} \ \ \forall w \in X^+;\ f_o^1(s_1, w) = f_o^2(s_2, w).$$

The set $\{p \in S_2 | (q,p) \in E(\mathcal{M}_1, \mathcal{M}_2)\}$ of all states of $\mathcal{M}_2$ that are equivalent to a state $q \in S_1$ of $\mathcal{M}_1$ is denoted by $[q]_{E(\mathcal{M}_1,\mathcal{M}_2)}$. When $\mathcal{M}_1 = \mathcal{M}_2 = \mathcal{M}$, the equivalence relation $E(\mathcal{M},\mathcal{M})$ partitions the state set S of $\mathcal{M}$ into the set of disjoint equivalence classes $S/E(\mathcal{M},\mathcal{M})$.

$\mathcal{M}_1$ and $\mathcal{M}_2$ are said to be *equivalent* if for every state $s_1 \in S_1$ there exists a state $s_2 \in S_2$ such that $(s_1, s_2) \in E(\mathcal{M}_1, \mathcal{M}_2)$, and vice-versa. If there exists a bijection $b_S : S_1 \to S_2$ satisfying:

- $\forall s \in S_1,\ \forall x \in X;\quad b_S(f_s^1(s,x)) = f_s^2(b_S(s),x)$ and $f_o^1(s,x) = f_o^2(b_S(s),x)$
- $b_S(s_0^1) = s_0^2$,

then $\mathcal{M}_1$ and $\mathcal{M}_2$ are said to be *isomorphic.* Isomorphic SMs can be considered identical since they differ only in names of states.

An *SM* is said to be *reduced* if no two of its states are equivalent to each other. Reduced SM equivalent to $\mathcal{M} = (X, Y, S, f_s, f_o, s_0)$ is $(X, Y, S/E(\mathcal{M},\mathcal{M}), f_s', f_o', [s_0]_{E(\mathcal{M},\mathcal{M})})$, with $f_s' : S/E(\mathcal{M},\mathcal{M}) \times X^* \to S/E(\mathcal{M},\mathcal{M})$ and $f_o' : S/E(\mathcal{M},\mathcal{M}) \times X^+ \to S/E(\mathcal{M},\mathcal{M})$ defined as follows:

$$\forall s \in S, \forall w \in X^*;\quad f_s'([s]_{E(\mathcal{M},\mathcal{M})}, w) = [f_s(s,w)]_{E(\mathcal{M},\mathcal{M})}, \tag{1}$$

$$\forall s \in S, \forall w \in X^+;\quad f_o'([s]_{E(\mathcal{M},\mathcal{M})}, w) = f_o(s,w). \tag{2}$$

3 Dynamical Systems

Analysis of dynamical systems (DSs) via state space structures plays an important role in experimenting and interpreting complex systems. Most of the important qualitative behaviors of a nonlinear system can be made explicit in the state space with a state space analysis. In this paper only discrete-time DSs (i.e. DSs evolving in discrete time) will be considered. Our theoretical knowledge about nonlinear DSs is far from complete. The state space of a nonlinear DS often consists of qualitatively different regions. It is useful to take into account the geometric information about the structures and spatial arrangements of these regions.

Among the most important characteristics of a DS are the fixed points, periodic orbits, their stability types, and the spatial arrangement of the corresponding stability regions. We review some of the basic concepts in DS theory.

A discrete-time DS can be represented as the iteration of a (differentiable, invertible) function $f : A \to A \quad (A \subseteq \Re^n)$, i.e.

$$x_{t+1} = f(x_t), \quad t \in \mathbf{Z}, \tag{3}$$

where $\mathbf{Z}$ denotes the set of all integers. For each $x \in A$, the iteration (3) generates a sequence of distinct points defining the orbit, or trajectory of x under f. Hence, the (forward) orbit of x under f is the set $\{f^m(x)|\ m \geq 0\}$. For $m \geq 1$, f^m is the composition of f with itself m times. f^0 is defined to be the identity map on A.

A point $x_* \in A$ is called a *fixed point of* f, if $f^m(x_*) = x_*$, for all $m \in \mathbf{Z}$. A point $x_* \in A$ is a *periodic point of* f, if $f^q(x_*) = x_*$ for some $q \geq 1$. The least such a value of q is called the *period of the point* x_* and the orbit of x_*. The set $\{x_*, f(x_*), ..., f^{q-1}(x_*)\}$ is said to be a *periodic orbit of* x_* of period q. Notice that a fixed point is a periodic point of period one, and a periodic point of f with period q is a fixed point of f^q. If x_* is a periodic point of period q for f, then so are all of the other points in the orbit of x_*.

Fixed and periodic points can be classified according to the behaviour of the orbits of points in their vicinity. A fixed point x_* is said to be asymptotically stable (or an *attractive point of* f), if there exists a neighborhood $O(x_*)$ of x_*, such that $\lim_{m \to \infty} f^m(x) = x_*$, for all $x \in O(x_*)$. As m increases, trajectories of points near to an asymptotically stable fixed point tend to it. The *basin of attraction* of an attractive fixed point x_* is the set $\{x \in A|\ \lim_{m \to \infty} f^m(x) = x_*\}$.

A fixed point x_* of f is asymptotically stable only if for each eigenvalue λ of $Df(x_*)$, the Jacobian of f at x_*, $|\lambda| < 1$ holds. The eigenvalues of $Df(x_*)$ govern whether or not the map f in a vicinity of x_* has contracting or expanding directions. Eigenvalues larger in absolute value than one lead to expansion, whereas eigenvalues smaller than one lead to contraction. If all the eigenvalues of $Df(x_*)$ are outside the unit circle, x_* is a *repulsive point*, or repellor. All points from a neighborhood of a repellor move away from it as m increases, or equivalently, move towards it as $-m$ decreases[3]. If some eigenvalues of $Df(x_*)$ are inside and some are outside the unit circle, x_* is said to be a *saddle point*. There is a set W^s of points x such that the trajectory of x tends to x_* for $m \to \infty$. W^s is called the *stable invariant manifold of* x_*. Similarly, the *unstable invariant manifold of* x_*, W^u, is the set of points x such that the trajectory of x tends to x_* for $m \to -\infty$.

[3] $f^{-m} = (f^{-1})^m$

Since any periodic point of period q can be thought of as a fixed point of f^q, these remarks apply to periodic points as well.

An *absorbing set of a set* $B \subseteq A$ *under the map* f is a set P such that for all $x \in B$, there exists $m_0 \geq 0$, for which $f^m(x) \in P$, for all $m \geq m_0$. For a given $x \in B$, the least such a value of m_0 is called the absorption level of x in P under the map f. An absorption region of P under the map f is defined as follows:

$$\mathcal{A}_f(P) = \{x \in A \mid \text{ there exists } m_0 \geq 0, \text{ such that } f^m(x) \in P, \text{ for all } m \geq m_0\}.$$

When $A \subseteq \Re$, or $A \subseteq \Re^2$, it is useful to code with colors (or different gray levels) the absorption levels of points from $\mathcal{A}_f(P)$ in P. We will refer to such a diagram as an *absorption diagram of* P *under the map* f.

$B \subseteq A$ is said to be *positively invariant set* of f if $f(B) \subseteq B$, i.e. trajectories of points from B stay in B. Trivially, A is positively invariant set of f, but in an effort to understand the dynamics of (3), we are usually interested in finding as minimal positively invariant set[4] as possible. If B is open and[5] $f(\overline{B}) \subset B$ then the set $\tilde{B} = \bigcap_{m \geq 0} f^m(B)$ is not only positively invariant, but also attracting, meaning that there is a neighborhood of $\tilde{B}$ such that all orbits starting in that neighborhood converge to $\tilde{B}$. Attractive fixed points and periodic orbits are trivial examples of attractive sets. Much more complicated attractive sets can be found in dynamical systems literature under the name strange attractors[6] [12]. As in the case of an attractive fixed point, the basin of attraction of an attractive set $\tilde{B}$ is the set of all points whose orbits converge to $\tilde{B}$.

If $B \subseteq A$ is positively invariant set of f then it is certainly an absorbing set of itself under f. B may be an attracting set of f, or it may contain an attractive set of f[7], or none of the two[8].

To learn more about the theory of DSs, see for example [19].

[4] in sense of inclusion

[5] $\overline{B}$ denotes the closure of B

[6] Loosely speaking, strange attractors are attractive sets that are topologically distinct from (i.e. cannot be transformed by a homeomorphism to) trivial attractive sets mentioned above.

[7] Note that this does not necessarily imply that B is part of basin of attraction of an attractive set contained in B. Think of attractive periodic orbit inside B that encircles a repelling fixed point.

[8] Identity map constitutes a simple example

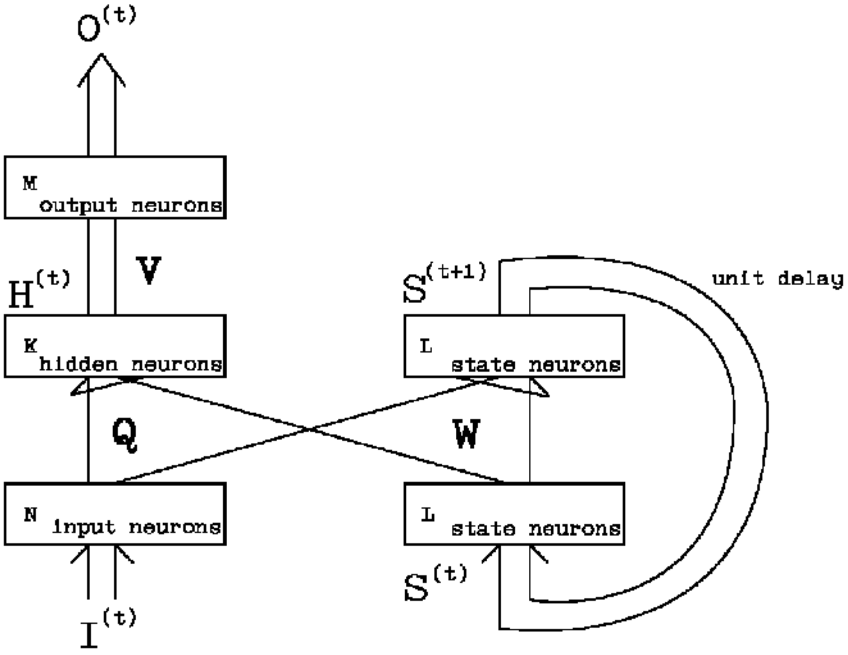

Figure 1: RNN model used for learning FSMs.

4 Recurrent Neural Network

The RNN presented in figure 1 was shown to be able to learn mappings that can be described by finite state machines [31]. A binary input vector $I^{(t)} = (I_1^{(t)}, ..., I_N^{(t)})$ corresponds to the activations of N input neurons. There are two types of hidden neurons in the network.

- K hidden nonrecurrent neurons H_1,...,H_K, activations of which are denoted by $H_j^{(t)}$, $j = 1, ..., K$.

- L hidden recurrent neurons S_1,...,S_L, called state neurons. We refer to the activations of state neurons by $S_i^{(t)}$, $i = 1, ..., L$. The vector $S^{(t)} = (S_1^{(t)}, ..., S_L^{(t)})$ is called the state of the network.

W_{iln}, Q_{jln} and V_{mk} are real-valued weights and g is a sigmoid function $g(x) = 1/(1+e^{-x})$. The activations of hidden nonrecurrent neurons are determined by

$$H_j^{(t)} = g(\sum_{l,n} Q_{jln} \cdot S_l^{(t)} \cdot I_n^{(t)}).$$

The activations of state neurons at the next time step $(t+1)$ are computed as follows:

$$S_i^{(t+1)} = g(\sum_{l,n} W_{iln} \cdot S_l^{(t)} \cdot I_n^{(t)}) = \mathcal{S}_i(S^{(t)}, I^{(t)}). \qquad (4)$$

The output of the network at time t is the vector $(O_1^{(t)}, ..., O_M^{(t)})$ of activations of M output neurons O_1,...,O_M. The network output is determined by

$$O_m^{(t)} = g(\sum_k V_{mk} \cdot H_k^{(t)}) = \mathcal{O}_m(S^{(t)}, I^{(t)}). \qquad (5)$$

Network states are elements of the L-dimensional open interval $(0,1)^L$, the internal region of the L-dimensional hypercube.

A unary encoding of symbols of both the input and output alphabets is used with one input and one output neuron for each input and output symbol respectively.

The bijection defining the encoding of N input symbols into N-dimensional binary vectors with just one active bit is denoted by c_I. Similarly, the bijection that defines the encoding of M output symbols into M-dimensional one-active-bit binary vectors is denoted by c_O.

The vector $I(t) = (I_1^{(t)}, ..., I_N^{(t)}) \in \{0,1\}^N$ of activations of input neurons corresponds to the input symbol $c_I^{-1}(I_1^{(t)}, ..., I_N^{(t)})$.

Activation of each output neuron is from the open interval $(0,1)$. A threshold $\Delta \in (0, \frac{1}{2})$ is introduced, such that any value from $(0, \Delta)$ is assumed to be an approximation of 0, and any value from $(1-\Delta, 1)$ represents the value 1. A mapping $r : (0,1) \rightarrow \{0, 1, -1\}$ is defined as follows[9]:

$$r(x) = \begin{cases} 0 & \text{if } x \in (0, \Delta) \\ 1 & \text{if } x \in (1-\Delta, 1) \\ -1 & \text{otherwise.} \end{cases}$$

[9] -1 represents *don't know* output of an output neuron

Interpretation of network output in terms of output symbols of the FSM it models is performed via mapping D[10]:

$$D(y_1,...,y_M) = \begin{cases} c_O^{-1}(y_1,...,y_M) & \text{if } \; y_i \in \{0,1\}, \quad i=1,...,M \\ * & \text{otherwise.} \end{cases}$$

If the output of the network, $O(t) = (O_1^{(t)},...,O_M^{(t)})$, falls into $((0,\Delta) \cup (1-\Delta,1))^M$, then it corresponds to the output symbol

$$D(r(O_1^{(t)}),...,r(O_M^{(t)})) = c_O^{-1}(r(O_1^{(t)}),...,r(O_M^{(t)})) = c_O^{-1}(R(O_1^{(t)},...,O_M^{(t)})) = c_O^{-1}(R(O^{(t)})),$$

where the map R is the component-wise application of the map r.

Each input word (a word over the input alphabet of the FSM used for training) is encoded into the input neurons one symbol per discrete time step t, yielding the corresponding output, as well as the network new state.

Training is performed via optimization with respect to the error function

$$E = \frac{1}{2}\sum_m (T_m^{(t)} - O_m^{(t)})^2,$$

where $T_m^{(t)} \in \{0,1\}$ is the desired response value for the m–th output neuron at the time step t. For a more detailed explanation of the training procedure see [31].

5 RNN as a State Machine

In this section we assume that a RNN $\mathcal{N}$ of the type described above has learned to exactly mimic the behaviour of a reduced, connected FSM $\mathcal{M} = (X, Y, Q, \delta, \lambda, s_o)$ it was trained with. It follows that there exists a network state S^0, for which network output will always be in $((0,\Delta)\cup(1-\Delta,1))^M$ upon presentation of any input word, and such that the following correspondence holds (time is set to $t = 1$):[11]

$$\forall w = x_1...x_n \in X^+; \quad \lambda(q_i, x_i) = D(R(O^{(i)})), \quad \text{for all } i = 1,...,n, \tag{6}$$

[10] It is assumed that * does not belong to the set of output symbols of the FSM modeled by the RNN. * stands for *don't know* output of the net.

[11] In practical terms, during learning phase, the network is trained to respond to a special "reset" input symbol # ($\# \notin X$) by changing its state to a state equivalent to s_0, the initial state of $\mathcal{M}$ (more details in [31]). S^0 is the "next-state" computed in the layer of recurrent state neurons when the symbol # is presented to the network input after training process has been completed.

where

- $q_1 = s_0$,
- $S^{(1)} = S^0$,
- $q_{i+1} = \delta(q_i, x_i),\ i = 1, ..., n-1$, and
- the network input $I^{(i)}$ at the time step i is the code $c_I(x_i)$ of the i-th input symbol x_i of the input word w.

Automata theory provides us with the ability to connect structural and behavioural equivalence of automata [32]. In particular, it can be shown, that for any couple $(\mathcal{M}_1, \mathcal{M}_2)$ of connected FSMs with equal input, as well as output sets it holds: if $B_{\mathcal{M}_1} = B_{\mathcal{M}_2}$, then $\mathcal{M}_1$ and $\mathcal{M}_2$ are equivalent and their reduced forms are isomorphic. To investigate the correspondence between $\mathcal{N}$ and $\mathcal{M}$ in this context, we represent the network $\mathcal{N}$ as a SM $\bar{\mathcal{N}} = (X, Y \cup \{*\}, \bar{S}, \tau, \nu, S^0)$, where the maps ν and τ are defined as follows:

$$\text{for any } S = (S_1, ..., S_L) \in \bar{S}, \quad \text{and any } x \in X;$$

$$\nu(S, x) = D(R(\mathcal{O}_1(S, c_I(x)), ..., \mathcal{O}_M(S, c_I(x)))),$$

and

$$\tau(S, x) = (\mathcal{S}_1(S, c_I(x)), ..., \mathcal{S}_L(S, c_I(x))),$$

with $\mathcal{O}_i$ and $\mathcal{S}_j$ defined by (5) and (4) respectively.

From (6) it follows that

$$\forall w \in X^+; \quad \lambda^+(s_0, w) = \nu^+(S^0, w). \tag{7}$$

The set $\bar{S} = (0,1)^L$ of states of $\bar{\mathcal{N}}$ can be partitioned into the set of equivalence classes corresponding to the equivalence relation $E(\bar{\mathcal{N}}, \bar{\mathcal{N}})$. By presenting inputs to the network and considering only the de-coded network outputs, it is impossible to distinguish between equivalent network states.

$[S^0]_{E(\bar{\mathcal{N}},\bar{\mathcal{N}})}$ is the set of all network states equivalent to S^0. Denote the set of network states accessible from states from $[S^0]_{E(\bar{\mathcal{N}},\bar{\mathcal{N}})}$ by $\bar{S}_{acc}$. Note that for every state $S \in \bar{S}_{acc}$ and for each input word $w \in X^+$, $\nu^+(S, w)$ does not contain the *don't know* symbol $*$. From $\bar{\mathcal{N}}$, a reduced, connected SM $\bar{\mathcal{N}}_1 = (X, Y, \bar{S}_{acc}/E(\bar{\mathcal{N}}, \bar{\mathcal{N}}), \tau_1, \nu_1, [S^0]_{E(\bar{\mathcal{N}},\bar{\mathcal{N}})})$ is constructed, where τ_1 and ν_1 are defined according to (1) and (2) respectively, and respectively restricted to $\bar{S}_{acc}/E(\bar{\mathcal{N}},\bar{\mathcal{N}}) \times X^*$ and $\bar{S}_{acc}/E(\bar{\mathcal{N}},\bar{\mathcal{N}}) \times X^+$. $\bar{\mathcal{N}}_1$ has the same behaviour as $\mathcal{M}$. It is easy to see that the number of states of $\bar{\mathcal{N}}_1$ is finite and hence $\bar{\mathcal{N}}_1$ is a FSM. It follows that $\bar{\mathcal{N}}_1$ and $\mathcal{M}$ are isomorphic.

The set $[q]_{E(\mathcal{M},\bar{\mathcal{N}})}$ of all network states equivalent to the state q of $\mathcal{M}$ is denoted by $(q)_{\mathcal{N}}$. States of a SM code the information about "what has happened so far in the course of input word processing". From that point of view, all network states from $(q)_{\mathcal{N}}$ code the same information, the information that is coded by the state q of $\mathcal{M}$.

So far we have dealt with the existence issues concerning nonempty regions of network states equivalent to states of the FSM the network is capable to exactly mimic. For a "constructive" approach to determination of $(q)_{\mathcal{N}}$, the regions $\mathcal{N}_x^y$ of network state space are identified, for which the network $\mathcal{N}$ gives the (decoded) output y provided the code of the input symbol x is presented at network input. In particular, $\mathcal{N}_x^y = \{S \in \bar{S} | \nu(S,x) = y\}$. Note that for each $x \in X$ and $y \in Y$, $\mathcal{N}_x^y$ is an open set. For a given input word $w = x_1 x_2 ... x_n \in X^+$, the set of all network states $\mathcal{N}_w^{\lambda^+(q,w)}$ originating the output equal to $\lambda^+(q,w)$ is

$$\mathcal{N}_w^{\lambda^+(q,w)} = \mathcal{N}_{x_1}^{\lambda(q,x_1)} \cap \left[\bigcap_{i=2}^{n} (\tau_{x_{i-1}} \circ ... \circ \tau_{x_2} \circ \tau_{x_1})^{-1} (\mathcal{N}_{x_i}^{\lambda(q,x_1 x_2 ... x_i)}) \right], \tag{8}$$

where

$$\tau_x(S) = \tau(S,x), \quad \text{for each } x \in X. \tag{9}$$

By $f^{-1}(A)$, where f is a map and A is a set, we denote the set of all points whose images under f are in A. For any $x \in X$, τ_x is continuous, and so is the composition $\tau_{x_m} \circ ... \circ \tau_{x_2} \circ \tau_{x_1}$ for any word $x_1 x_2 ... x_m \in X^+$. It follows that the sets $\mathcal{N}_w^{\lambda^+(q,w)}$ are open. However, the set

$$(q)_{\mathcal{N}} = \bigcap_{w \in X^+} \mathcal{N}_w^{\lambda^+(q,w)} \tag{10}$$

of network states equivalent to the state q of $\mathcal{M}$ is not necessarily open, since an infinite, countable intersection of open sets is not guaranteed to be open[12]. If $(q)_{\mathcal{N}}$ is open, $(q)_{\mathcal{N}} \neq \emptyset$ implies there exists a (finite) length L of input words such that[13] $(q)_{\mathcal{N}} = \bigcap_{|w| \leq L} \mathcal{N}_w^{\lambda^+(q,w)}$.

[12] The case when trajectories in the RNN state space may be corrupted by a noise is not discussed in this paper. However, we note that if $(q)_{\mathcal{N}}$ is not open, arbitrarily close to a state $S \in (q)_{\mathcal{N}}$ there is a network state not equivalent to the state q of $\mathcal{M}$ and an arbitrarily small perturbation of S may cause failure in the RNN modeling of $\mathcal{M}$.

[13] $|w|$ denotes length of the word w, i.e. the number of symbols contained in w

From (8) and (10) it follows that if there is an x-loop in a state q of $\mathcal{M}$ producing an output symbol y, then

$$\tau_x((q)_{\mathcal{N}}) \subseteq (q)_{\mathcal{N}} \subseteq \bigcap_{i \geq 0} (\tau_x^i)^{-1}(\mathcal{N}_x^y). \tag{11}$$

As in section 3, τ_x^i is the composition of τ_x with itself i times. τ_x^0 is defined to be the identity map.

Analogically, if there is an x-cycle of length m passing through states $q_1, ..., q_m$ with outputs $y_i = \lambda(q_i, x),\ i = 1, ..., m$, then

$$(q_1)_{\mathcal{N}} \subseteq \bigcap_{j=1}^{m} (\tau_x^{j-1})^{-1} \left(\bigcap_{i \geq 0} (\tau_x^{im})^{-1}(\mathcal{N}_x^{y_j}) \right). \tag{12}$$

Similar bounds can be found for $(q_2)_{\mathcal{N}}, ..., (q_m)_{\mathcal{N}}$, in particular

$$\tau_x^m((q_j)_{\mathcal{N}}) \subseteq (q_j)_{\mathcal{N}} \subseteq \bigcap_{i \geq 0} (\tau_x^{im})^{-1}(\mathcal{N}_x^{y_j}), \quad j = 1, ..., m. \tag{13}$$

Some researchers attempted to extract learned automaton from a trained recurrent network [17], [8], [37], [31]. Extraction procedures rely on the assumption that equivalent network states are grouped together in well-separated regions in the recurrent neurons' activation space. After training, the network state space is partitioned into clusters using some clustering tool and for each $q \in Q$, the region $(q)_{\mathcal{N}}$ is approximated by (possibly) several of such obtained clusters. For example, in [17] the network state neurons' activation space is divided into several equal hypercubes. When the number of hypercubes is sufficiently high, each hypercube is believed to contain only mutually equal states. After training, Tiňo and Šajda [31] present a large number of input words to the network input. All states the network passes through during the presentation are saved. Then the clustering of those states is performed using Kohonen map with "star" topology of neural field consisting of several "branches" of neurons connected to one "central" neuron. Such a topology helped to reduce great sensitivity to initial conditions found in vector-coding algorithms using independent cluster centers, while avoiding time consuming approximation of input space topology typical of classical regular-grid topologies of Kohonen Map [30]. Other approaches to RNN state space clustering are discussed in [31].

Having approximated the regions $(q)_{\mathcal{N}}$, the automaton $\bar{\mathcal{N}}_1$ is constructed via determining arcs in the corresponding transition diagram, followed by non-determinism eliminating and minimization procedures.

All ideas presented in this section stem from the assumption, that the network $\mathcal{N}$ exactly mimics the FSM $\mathcal{M}$ it was trained with. However, it is possible that a correct automaton is extracted from trained RNN even though the network is known to generalize poorly on long, unseen input words [17]. This is discussed in section 8.

5.1 Experiments

Number of experiments were performed in which RNNs with two or three state neurons were trained simple FSMs. To show how the network learned to organize its state space in order to mimic a given FSM, the regions corresponding to $(q)_{\mathcal{N}}$ were detected. The network state space was "covered" with a regular grid $\mathcal{G}$ of $R \times R$ points (R is of order of hundreds) and a finite vocabulary Γ of distinguishing sequences of $\mathcal{M}$ was created. Regions $(q)_{\mathcal{N}}$ were approximated by grouping together those network states from the grid that, for each input word from the vocabulary, lead to equal output strings. In other words, $(q)_{\mathcal{N}} = \bigcap_{w \in X^+} \mathcal{N}_w^{\lambda^+(q,w)}$ were approximated by $\bigcap_{w \in \Gamma} \mathcal{N}_w^{\lambda^+(q,w)} \cap \mathcal{G}$. For example, in figure 3 approximations of regions of equivalent network states corresponding to states of a FSM shown in figure 2 can be seen. Figure 3 should be compared with figure 4 showing activations of state neurons during presentation of training set to the RNN after training.

Generally, in our experiments, regions approximating $(q)_{\mathcal{N}}$ were observed to be connected and of "simple shape". Further study needs to be devoted to that matter. However, at least empirically and for simple tasks, our use of the Kohonen Map as a clustering tool [31], as well as the use of simple clustering technique introduced in [17] are supported.

6 RNN as a Collection of Dynamical Systems

RNNs can be viewed as discrete-time DSs. Literature dealing with the relationship between RNNs and DSs is quite rich: [20], [3], [16], [6], [7] [24], [26], [35], [36], [34], [2], [21], for example. However, as it has been already mentioned, the task of complete understanding of the global dynamical behaviour of a given DS is not at all an easy one. In [36] it is shown that networks with just two recurrent neurons can exhibit chaos and hence the asymptotic network dynamical behaviour (on a chaotic attractor) can be very complex.

In order to describe the behaviour of the RNN $\mathcal{N}$ by an iterative map, we confine ourselves

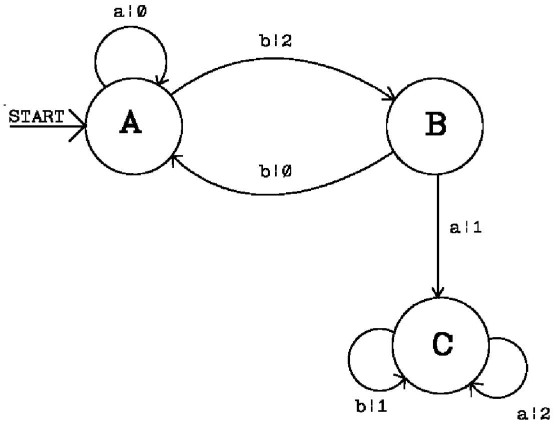

Figure 2: FSM $\mathcal{M}$ used for training RNN. $\mathcal{M} = (X, Y, S, f_s, f_o, s_0)$ is represented as a directed graph called the state transition diagram. The graph has node for each state, and every node has $|X|$ ($|X|$ denotes the number of elements of a finite set X) outgoing arcs labeled with $x|y$ ($x \in X$, $y \in Y$) according to the rule: The arc from the node labeled with $s_1 \in S$ to the node labeled with $s_2 \in S$ is labeled with $x|y$ if $s_2 = f_s(s_1, x)$, *and* $y = f_o(s_1, x)$. The node corresponding to the initial state is indicated by an arrow labeled with START.

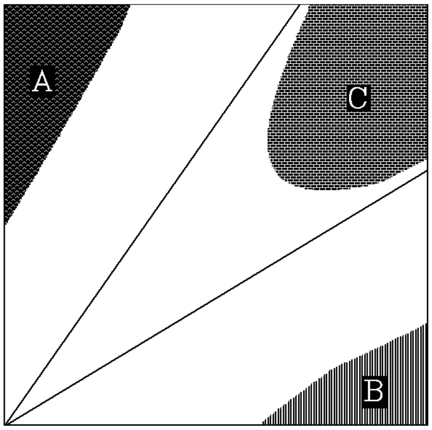

Figure 3: Regions of equivalent network states. Capital letter inside each region indicates to which state of $\mathcal{M}$ the network states from that region are equivalent. $\Delta = 0.1$. Two lines stemming from the origin are the lines $\tau_a(s)_1 = 1/2$ and $\tau_a(s)_2 = 1/2$, between them is the region $\mathcal{P}_{a,(1,1)}$ (see section 6).

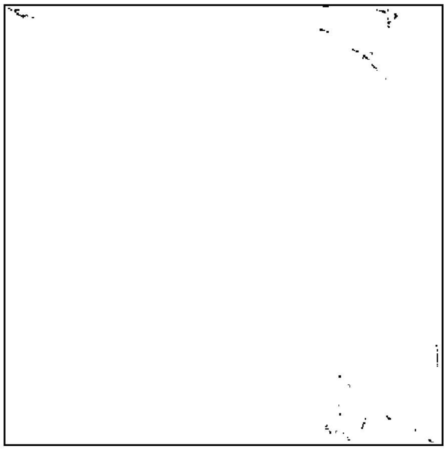

Figure 4: Activations of state neurons when training set is presented to the network after training process has finished (weights are frozen).

to only one input symbol x from the input alphabet of the FSM used for training $\mathcal{N}$, the code of which is repeatedly presented to the network input. The evolution of the network is described in terms of trajectories $\{S, \tau_x(S), \tau_x^2(S), ...\}$ in $(0,1)^L$. The iterative map $\tau_x : (0,1)^L \rightarrow (0,1)^L$ is defined in (9).

As in the previous section, here we also assume that a RNN $\mathcal{N}$ exactly mimics the behaviour of a reduced, connected FSM $\mathcal{M}=(X, Y, Q, \delta, \lambda, s_o)$. In this section we deal with the problem of how certain features of $\mathcal{M}$ found in its STD (such as loops and cycles) induce some specific features (such as attractive points and periodic orbits) of network global dynamical behaviour.

Assume that there is an x-loop in a state q of $\mathcal{M}$ and $\lambda(q,x) = y$. Then according to (11), $(q)_{\mathcal{N}}$ is a positively invariant set of τ_x and hence an absorbing set of itself under τ_x. From (8) it follows that, under τ_x, $(q)_{\mathcal{N}}$ is an absorbing set of all sets $(p)_{\mathcal{N}}$ such that q is x-accessible from p. If there is an open set B such that $B \subseteq (q)_{\mathcal{N}}$ and $\tau_x(\overline{B}) \subset B$, or $(q)_{\mathcal{N}} \subseteq B$ and $\tau_x(\overline{B}) \subset (q)_{\mathcal{N}}$, then there is an attractive set $\bigcap_{m \geq 0} f^m(B)$ of τ_x in $(q)_{\mathcal{N}}$ that constitutes a stable network representation of the x-loop in a state q of $\mathcal{M}$.

Similarly, assume that there is an x-cycle γ of length m passing through states $q_1, ..., q_m$ with outputs $y_j = \lambda(q_j, x)$, $j = 1, ..., m$. Then according to (13), $(q_j)_{\mathcal{N}}$ are positively invariant sets of τ_x^m and $\bigcup_{j=1}^m (q_j)_{\mathcal{N}}$ is positively invariant set of τ_x. A statement concerning the existence of attractive sets of τ_x^m inside $(q_j)_{\mathcal{N}}$ (or an attractive set of τ_x inside $\bigcup_{j=1}^m (q_j)_{\mathcal{N}}$) can be made analogically to the statement above. Considering (8) it can be seen that under τ_x, $\bigcup_{q\in\gamma}(q)_{\mathcal{N}}$ is an absorbing set of itself and all sets $(p)_{\mathcal{N}}$ such that γ is x-accessible from p.

Observation 1 formulates these ideas in a more compact form.

Observation 1: *Assume that a RNN $\mathcal{N}$ exactly mimics the behaviour of a reduced, connected FSM $\mathcal{M}=(X,Y,Q,\delta,\lambda,s_o)$. Then*

- *if there is an x-loop in a state q of $\mathcal{M}$, then $(q)_{\mathcal{N}} \subseteq \mathcal{N}_x^{\lambda(q,x)}$ is positively invariant set of τ_x and* [14] $\bigcup_{q\in Acc(x,p)}(p)_{\mathcal{N}} \subseteq \mathcal{A}_{\tau_x}\left((q)_{\mathcal{N}}\right)$.
- *if there is an x-cycle γ of length m passing through states $q_1,...,q_m$ of $\mathcal{M}$, then $(q_j)_{\mathcal{N}}$, $j=1,...,m$ are positively invariant sets of τ_x^m and $\bigcup_{j=1}^m(q_j)_{\mathcal{N}}$ is positively invariant set of τ_x. $(q_1)_{\mathcal{N}},...,(q_m)_{\mathcal{N}}$ are periodically visited in the process of iteration of τ_x, and $\bigcup_{\gamma\subseteq Acc(x,p)}(p)_{\mathcal{N}} \subseteq \mathcal{A}_{\tau_x}\left(\bigcup_{q\in\gamma}(q)_{\mathcal{N}}\right)$.*

When there was an x-loop in a state q of $\mathcal{M}$ in all our experiments an attractive fixed point S_* of τ_x "near" a vertex $v \in \{0,1\}^L$ was detected (see subsection Experiments bellow). If $S_* \in (q)_{\mathcal{N}}$, S_* constitutes a plausible network representation of the x-loop. If furthermore S_* is the only attractive set of τ_x inside $(q)_{\mathcal{N}}$, then $\bigcup_{q\in Acc(x,p)}(p)_{\mathcal{N}}$ is a subset of its basin of attraction.

For each input symbol x of $\mathcal{M}$ and each vertex $v=(v_1,...,v_L)\in\{0,1\}^L$ define the set [15]

$$\mathcal{P}_{x,v} = \left\{ s\in\Re^L \mid \tau_x(s)_i < \frac{1}{2} \text{ if } v_i = 0; \quad \tau_x(s)_i > \frac{1}{2} \text{ if } v_i = 1; \quad i=1,...,L \right\}.$$

Hyperplanes $\tau_x(s)_i = 1/2$ separate $\Re^L$ into 2^L partitions $\mathcal{P}_{x,v}$. The map τ_x is transformed to the map τ_x^μ by multiplying weights W_{iln} by a scalar $\mu > 0$, i.e. $\tau_x^\mu(s) = \tau_x(\mu s)$. μ is also called the *neuron gain.* The following Lemma was proved by Li [27]. It is stated for maps τ_x and accommodated with our notation. It tells us under what conditions one may expect an attractive fixed point of τ_x^μ to exist "near" a vertex $v\in\{0,1\}^L$.

Lemma 1: (Li, 1992) *Suppose that for some input symbol x of $\mathcal{M}$ there exists a vertex $v \in \overline{\mathcal{P}_{x,v}\cap\tau_x(\mathcal{P}_{x,v})}$. Then there exists a neuron gain μ_0 such that for all $\mu>\mu_0$ there is an attractive fixed point of τ_x^μ in $\mathcal{P}_{x,v}\cap\tau_x(\mathcal{P}_{x,v})$.*

[14] recall that $\mathcal{A}_{\tau_x}\left((q)_{\mathcal{N}}\right)$ is the absorbing region of $(q)_{\mathcal{N}}$ under map τ_x

[15] $\tau_x(s)_i$ denotes the i-th component of $\tau_x(s)$. When viewed as an iterative map, τ_x operates on $(0,1)^L$, but here we allow $s\in\Re^L$.

It was also shown that as μ tends to infinity, the attractive fixed point tends to the vertex v. For two recurrent neurons, under certain conditions on weights W_{iln}, this is made more specific in the next section (Corollary 1).

Theorem 1: *In addition to the assumptions in Observation 1, assume there is an x-loop in a state q of $\mathcal{M}$. Suppose there is a vertex $v \in \{0,1\}^L$ such that $(q)_{\mathcal{N}} \subseteq \mathcal{P}_{x,v}$ and $v \in \overline{\tau_x((q)_{\mathcal{N}})}$. Then there exists a neuron gain μ_0 such that for all $\mu > \mu_0$ there exists an attractive fixed point $S_* \in \mathcal{P}_{x,v} \cap \tau_x(\mathcal{P}_{x,v})$ of τ_x^μ.*

Proof: From

$$\tau_x((q)_{\mathcal{N}}) \subseteq (q)_{\mathcal{N}} \subseteq \mathcal{P}_{x,v} \quad \text{and} \quad \tau_x((q)_{\mathcal{N}}) \subseteq \tau_x(\mathcal{P}_{x,v})$$

it follows that $\tau_x((q)_{\mathcal{N}}) \subseteq \mathcal{P}_{x,v} \cap \tau_x(\mathcal{P}_{x,v})$. Hence

$$v \in \overline{\tau_x((q)_{\mathcal{N}})} \subseteq \overline{\mathcal{P}_{x,v} \cap \tau_x(\mathcal{P}_{x,v})}.$$

Employing Lemma 1, the result follows immediately. □

Loosely speaking, Theorem 1 says that if arbitrarily close to a vertex $v \in \{0,1\}^L$ there is a network state from $\tau_x((q)_{\mathcal{N}}) \subseteq (q)_{\mathcal{N}} \subseteq \mathcal{P}_{x,v}$, i.e. if network states that are equivalent to the state q of $\mathcal{M}$ in which there is an x-loop are "accumulated" around the vertex v within $\mathcal{P}_{x,v}$, then if the weights are "large enough", so that $\mu_0 < 1$, an attractive fixed point of τ_x exists in a neighborhood of v (figures 3 and 5).

As mentioned in the introduction, the approach presented in [6] addresses representational issues concerning recurrent neural networks trained to act as regular language recognizers. Recurrent neural networks are assumed to operate in a noisy environment. Such an assumption can be supported by an argument that in any system implemented on a digital computer there is a finite amount of noise due to round-off errors and "*we are only interested in solutions wich work in spite of round-off errors*" [6]. Orbits of points under a map f and attractive sets of f are substituted for by the notions of ϵ-pseudo-orbit of points under f and ϵ-pseudo-attractor of f. These concepts correspond to the idea that instead of the *precise* trajectory of a point under a map we should consider each sequence of points (pseudotrajectory) having the distance from the precise trajectory less than $\epsilon > 0$. It is proved that when there is a loop in the reduced acceptor of a regular language also recognized by the network, then there must be an ϵ-pseudo-attractor (and hence an attractor) of the corresponding map in the network state space. The network accepts and rejects a string of symbols if ϵ-pseudo-orbits driven by the string end in subregions denoted by accept and reject regions respectively. It is assumed that the accept and reject regions are closed in the network state space.

6.1 Experiments

To see how loops and cycles of a FSM $\mathcal{M}$ are transformed into global dynamical properties of a RNN $\mathcal{N}$ that is able to exactly mimic $\mathcal{M}$, the following experiments were performed:

Consider again the FSM $\mathcal{M}$ presented in figure 2. In figure 3 it can be seen how the RNN $\mathcal{N}$ with two state neurons organizes its state space, $(0,1)^2$, into three distinct, connected regions $(A)_\mathcal{N}$, $(B)_\mathcal{N}$, and $(C)_\mathcal{N}$, corresponding to states A, B, and C respectively. It was observed[16] that trajectories starting in $(A)_\mathcal{N}$ converged to a single attractive point placed inside $(A)_\mathcal{N}$. The same applies to the state C, and its corresponding region $(C)_\mathcal{N}$. So the a-loops in the states A and C induce attractive points of τ_a placed inside the corresponding regions of equivalent RNN states. Actually, this represents the only RNN stable representation of loops in $\mathcal{M}$ we have observed during our simulations.

$(A)_\mathcal{N}$ and $(C)_\mathcal{N}$ are absorbing sets of themselves under the map τ_a. Since the state C is a-accessible from B, $(C)_\mathcal{N}$ is an absorbing set of $(B)_\mathcal{N}$ under τ_a. Absorption diagrams of $(A)_\mathcal{N}$ and $(C)_\mathcal{N}$ under τ_a together with the attractive points are presented in figure 5.

If we presented $\mathcal{M}$ only with input symbol b, we would end up either in a b-cycle of length two involving states A and B, or in a b-loop in the state C. When, during the experiments, we started in a state from $(C)_\mathcal{N}$, and presented to the network input only the code of the symbol b, the trajectory converged to an attractive point inside $(C)_\mathcal{N}$. An absorption diagram of $(C)_\mathcal{N}$ under τ_b together with the attractive point can be seen in figure 6.

On the other hand, when started in a state from $(A)_\mathcal{N}$, the trajectory jumped between the sets $(A)_\mathcal{N}$ and $(B)_\mathcal{N}$ converging to a periodic orbit of length two. Again, this was observed to be the typical stable RNN representation of a cycle corresponding to an input symbol of $\mathcal{M}$. The states

[16] As before, during the simulations, the network state space was "covered" with a regular grid of points and only the orbits starting from these points were taken into account.

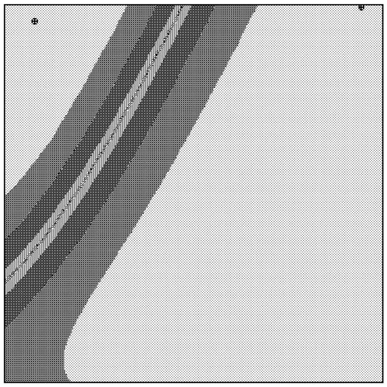

Figure 5: Absorption diagrams of $(A)_{\mathcal{N}}$ and $(C)_{\mathcal{N}}$ under the map τ_a. Network states lying in the lightest region need one or no iteration step under the map G_a to get to their absorption set. The more iteration steps are needed, the darker the region is, with the exception of the region "close to" the "border line" between the two absorption diagrams. The region is light so that the border contours are clearly visible. The figure should be compared with the figure in the previous section showing $(A)_{\mathcal{N}}$ and $(C)_{\mathcal{N}}$. Note the two attractive points of τ_a placed inside $(A)_{\mathcal{N}}$ and $(C)_{\mathcal{N}}$ induced by a-loops in states A and C respectively.

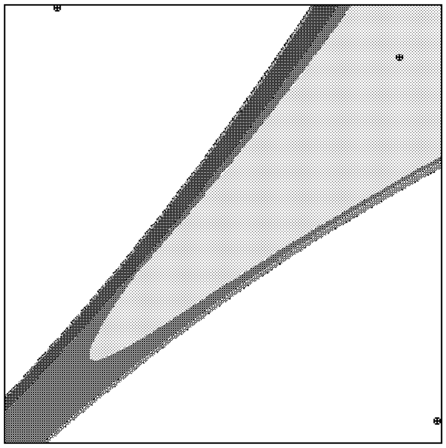

Figure 6: Absorption diagram of $(C)_{\mathcal{N}}$ under the map τ_b. Network states from the two white regions do not belong to the absorption region of $(C)_{\mathcal{N}}$. The figure should be compared with the figure in the previous section showing $(C)_{\mathcal{N}}$. Note the attractive point of τ_b placed inside $(C)_{\mathcal{N}}$ induced by the b-loop in the state C, as well as, two periodic points of τ_b placed inside $(A)_{\mathcal{N}}$ and $(B)_{\mathcal{N}}$ constituting an attractive periodic orbit of period two. The orbit is induced by the b-cycle $\{A, B\}$.

constituting the orbit can be seen in figure 6.

In the second experiment, a FSM $\mathcal{M}$ shown in figure 7 was used to generate the training set for a RNN $\mathcal{N}$ with three state neurons. The a-cycle $\{A, B, C, D, E\}$ of length five induced an attractive periodic orbit of τ_a of period five. Projections of the orbit to a two–dimensional subspace $(0,1)^2$ of the network state space can be seen in figures 8, 9, 10. To illustrate the convergence of orbits, the orbits were plotted after 60, 100, and 300 pre-iterations (figures 8, 9, and 10 respectively). No plotting occurred during the pre-iterations.

7 RNN with Two State Neurons

Usually, studies of the asymptotic behaviour of recurrent neural networks assume some form of structure in the weight matrix describing connectivity pattern among recurrent neurons. For example, symmetric connectivity and absence of self-interactions enabled Hopfield [22] to interpret the network as a physical system having energy minima in attractive fixed points of the network. These rather strict conditions were weakened in [7], where a more easily satisfied conditions are

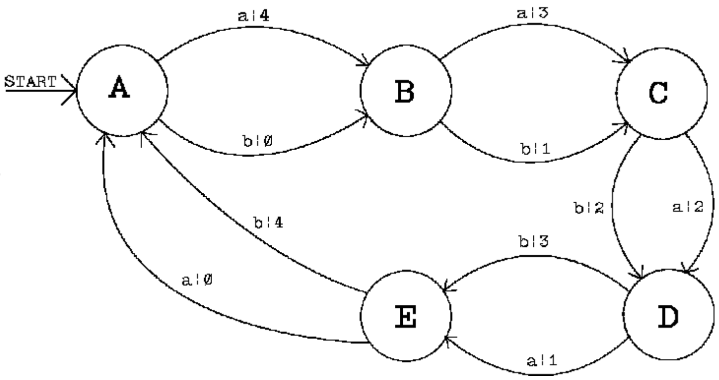

Figure 7: FSM $\mathcal{M}$ whose state transition diagram contains cycle of length five.

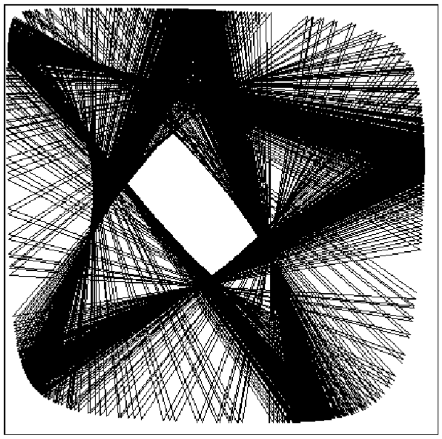

Figure 8:

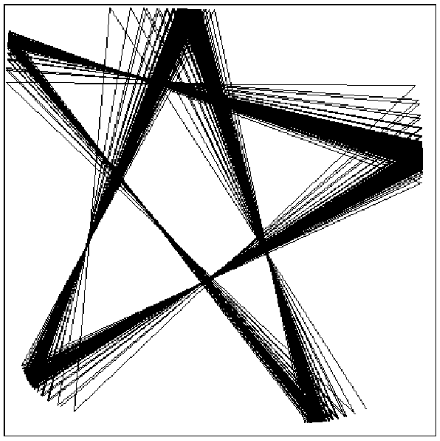

Figure 9:

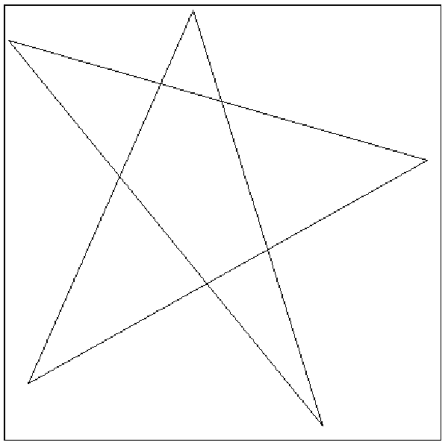

Figure 10:

formulated. Blum and Wang [3] globally analyze networks with nonsymmetrical connectivity patterns of special types. In case of two recurrent neurons with sigmoidal activation function g, they give results for weight matrices with diagonal elements equal to zero[17]. Recently, Jin, Nikifiruk and Gupta [25] reported new results on the absolute stability for a rather general class of recurrent neural networks. Conditions under which all fixed points of the network are attractive were determined by the weight matrix of the network.

The purpose of this section is to investigate the position and stability types of fixed points of maps τ_x under certain assumptions concerning the signs and magnitudes of weights W_{iln}. The iterative map under consideration can be written as follows:

$$(u_{n+1}, v_{n+1}) = (g(\alpha u_n + \beta v_n), g(\gamma u_n + \delta v_n)), \tag{14}$$

where $(u_n, v_n) \in (0,1)^2$ is the state of recurrent network with two state neurons at the time step n, and α, δ and β, γ are positive and negative real coefficients respectively. Thus we investigate the case when the two recurrent neurons are self-exciting ($\alpha, \delta > 0$), with the tendency to inhibit each other ($\beta, \gamma < 0$).

For $c > 4$, define

$$\Delta(c) = \frac{1}{2}\sqrt{1 - \frac{4}{c}}$$

In the following it will be shown how the network state space $(0,1)^2$ can be partitioned into regions

[17] In such a case the recurrent network is shown to have only one fixed point and no “genuine” periodic orbits (of period greater than one)

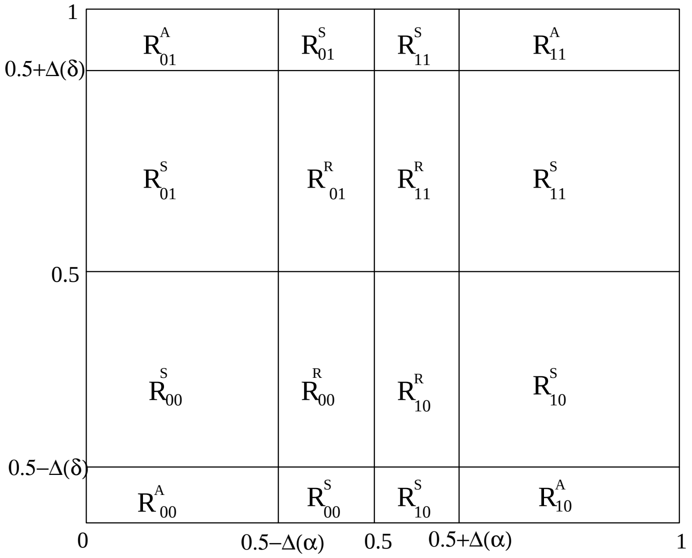

Figure 11: Partitioning of RNN state space according to stability types of fixed points of maps τ_x.

according to the stability types of fixed points of (14) found in the regions.

Regions

$$\left(0, \frac{1}{2} - \Delta(\alpha)\right) \times \left(0, \frac{1}{2} - \Delta(\delta)\right),$$

$$\left(\frac{1}{2} - \Delta(\alpha), \frac{1}{2}\right] \times \left(0, \frac{1}{2} - \Delta(\delta)\right) \cup \left(0, \frac{1}{2} - \Delta(\alpha)\right) \times \left(\frac{1}{2} - \Delta(\delta), \frac{1}{2}\right]$$

and

$$\left(\frac{1}{2} - \Delta(\alpha), \frac{1}{2}\right] \times \left(\frac{1}{2} - \Delta(\delta), \frac{1}{2}\right]$$

are denoted by R^A_{00}, R^S_{00} and R^R_{00} respectively. Regions symmetrical to R^A_{00}, R^S_{00} and R^R_{00} with respect to the line $u = 1/2$ are denoted by R^A_{10}, R^S_{10} and R^R_{10} respectively:

$$R^A_{10} = \left(\frac{1}{2} + \Delta(\alpha), 1\right) \times \left(0, \frac{1}{2} - \Delta(\delta)\right),$$

$$R^S_{10} = \left[\frac{1}{2}, \frac{1}{2} + \Delta(\alpha)\right) \times \left(0, \frac{1}{2} - \Delta(\delta)\right) \cup \left(\frac{1}{2} + \Delta(\alpha), 1\right) \times \left(\frac{1}{2} - \Delta(\delta), \frac{1}{2}\right],$$

$$R^R_{10} = \left[\frac{1}{2}, \frac{1}{2} + \Delta(\alpha)\right) \times \left(\frac{1}{2} - \Delta(\delta), \frac{1}{2}\right].$$

Similarly, let R^A_{01}, R^S_{01} and R^R_{01} denote the regions symmetrical to R^A_{00}, R^S_{00} and R^R_{00} with respect to the line $v = 1/2$. Finally, R^A_{11}, R^S_{11} and R^R_{11} denote regions that are symmetrical to R^A_{01}, R^S_{01} and R^R_{01} with respect to the line $u = 1/2$ (figure 11).

Theorem 2: *Suppose $\alpha > 4, \beta < 0, \gamma < 0, \delta > 4, \alpha > |\beta|, \delta > |\gamma|$. Then the following can be said about the fixed points of (14):*

- *attractive and repulsive points can lie only in $\bigcup_{i\in\mathcal{I}} R_i^A$ and $\bigcup_{i\in\mathcal{I}} R_i^R$ respectively. $\mathcal{I}$ is the index set $\mathcal{I} = \{00, 10, 01, 11\}$. If $max\{\alpha(\delta-4), \delta(\alpha-4)\} < \beta\gamma$, there are no repellors.*
- *all fixed points in $\bigcup_{i\in\mathcal{I}} R_i^S$ are saddle points*[18].

Proof: Any fixed point (u, v) of (14) satisfies

$$(u, v) = (g(\alpha u + \beta v), g(\gamma u + \delta v)). \tag{15}$$

Jacobian $J(u, v)$ of (14) in (u, v) is given by

$$\begin{pmatrix} \alpha G_1(u,v) & \beta G_1(u,v) \\ \gamma G_2(u,v) & \delta G_2(u,v) \end{pmatrix},$$

where $G_1(u, v) = g'(\alpha u + \beta v)$ and $G_2(u, v) = g'(\gamma u + \delta v)$. Since $g'(p) = g(p)(1 - g(p))$, considering (15) we have

$$(G_1(u, v), G_2(u, v)) = (u(1-u), v(1-v)) = \phi(u, v). \tag{16}$$

The eigenvalues of J are[19]

$$\lambda_{1,2} = \frac{\alpha G_1 + \delta G_2 \pm \sqrt{D}}{2},$$

where $D = (\alpha G_1 - \delta G_2)^2 + 4G_1 G_2 \beta\gamma$.

D is always positive and so is $\alpha G_1 + \delta G_2$. It follows that to identify possible values of G_1 and G_2 so that $|\lambda_{1,2}| < 1$, it is sufficient to solve the inequality $\alpha G_1 + \delta G_2 + \sqrt{D} < 2$, or equivalently

$$2 - \alpha G_1 - \delta G_2 > \sqrt{D}. \tag{17}$$

Consider only G_1, G_2 such that $\alpha G_1 + \delta G_2 < 2$, that is, (G_1, G_2) lies under the line $\rho : \alpha G_1 + \delta G_2 = 2$. All (G_1, G_2) above ρ lead to at least one eigenvalue of J greater than 1. Squaring both sides of (17) we arrive at

$$(\alpha\delta - \beta\gamma)G_1 G_2 - \alpha G_1 - \delta G_2 > -1. \tag{18}$$

[18] Note that this does not exclude the existence of saddle fixed points in other regions.

[19] to simplify the notation, the identification (u, v) of a fixed point in which (14) is linearized is omitted

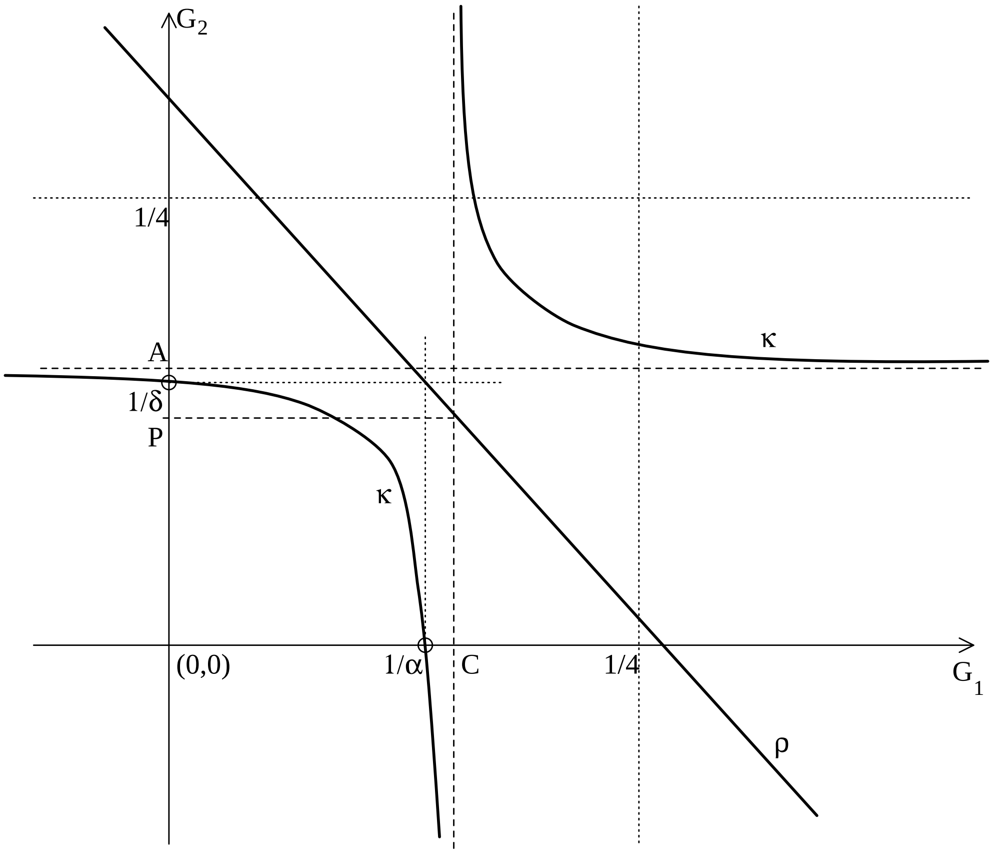

Figure 12:

The "border" curve κ : $(\alpha\delta - \beta\gamma)G_1G_2 - \alpha G_1 - \delta G_2 = -1$ in (G_1, G_2)-space is a hyperbola $G_2 = \kappa(G_1) = A[1 + B/(G_1 - C)]$, where

$$A = \frac{1}{\delta - \frac{\beta\gamma}{\alpha}}, \quad C = \frac{1}{\alpha - \frac{\beta\gamma}{\delta}}, \quad \text{and} \quad B = C - \frac{1}{\alpha}.$$

Since $0 < \delta - \beta\gamma/\alpha < \delta$ and $0 < \alpha - \beta\gamma/\delta < \alpha$, it follows that $A > 1/\delta, C > 1/\alpha$ and $B > 0$. $\kappa(1/\alpha) = 0, \kappa(0) = 1/\delta$ and (G_1, G_2) satisfying (18) lie under the "left branch" and above the "right branch" of κ (see figure 12). It is easy to see that since we are confined to the space below the line ρ, only (G_1, G_2) under the left branch of κ will be considered. Indeed, ρ is a decreasing line going through (C, P) and $A - P = 2(A - 1/\delta) > 0$, so it never intersects the right branch of κ.

A necessary (not sufficient) condition for a fixed point (u, v) of (14) to be attractive is that the corresponding $(G_1, G_2) = \phi(u, v) \in (0, 1/4]^2$ lies in $(0, 1/\alpha) \times (0, 1/\delta)$, where the map ϕ is defined by (16). For each $(G_1, G_2) \in (0, 1/4]^2$, under ϕ, there are four preimages

$$(u, v) = \phi^{-1}(G_1, G_2) = \left\{ \left(\frac{1}{2} \pm \Delta\left(\frac{1}{G_1}\right), \frac{1}{2} \pm \Delta\left(\frac{1}{G_2}\right) \right) \right\}. \tag{19}$$

The set of preimages of $(0, 1/\alpha) \times (0, 1/\delta)$ is the set $\bigcup_{i \in \mathcal{I}} R_i^A$, $\mathcal{I} = \{00, 10, 01, 11\}$.

A fixed point (u, v) of (14) is a saddle if $|\lambda_2| < 1$ and $|\lambda_1| = \lambda_1 > 1$. Since $\alpha\delta > \beta\gamma$,

$$0 < \sqrt{(\alpha G_1 + \delta G_2)^2 - 4G_1G_2(\alpha\delta - \beta\gamma)} = \sqrt{D} < \alpha G_1 + \delta G_2.$$

It follows that if $\alpha G_1 + \delta G_2 < 2$, i.e. (G_1, G_2) lies under the line ρ, $0 < \alpha G_1 + \delta G_2 - \sqrt{D} < 2$ holds and $0 < \lambda_2 < 1$. For (G_1, G_2) above the line ρ, i.e. $\alpha G_1 + \delta G_2 > 2$, we solve the inequality $\alpha G_1 + \delta G_2 - 2 < \sqrt{D}$, that leads to the "border" curve $G_2 = \kappa(G_1)$ we have already described. This time, only (G_1, G_2) "between" the two branches of hyperbola κ are considered.

It can be seen that in all fixed points (u, v) of (14) with

$$\phi(u, v) \in \left(0, \frac{1}{4}\right] \times \left(0, min\left\{A, \frac{1}{4}\right\}\right) \cup \left(0, min\left\{C, \frac{1}{4}\right\}\right) \times \left(0, \frac{1}{4}\right],$$

the eigenvalue $\lambda_2 > 0$ is less than 1. This is certainly true for all (u, v) such that $\phi(u, v) \in (0, 1/4] \times (0, 1/\delta) \cup (0, 1/\alpha) \times (0, 1/4]$. In particular, the preimages of $(G_1, G_2) \in (1/\alpha, 1/4] \times (0, 1/\delta) \cup (0, 1/\alpha) \times (1/\delta, 1/4]$ under ϕ define the region $\bigcup_{i \in \mathcal{I}} R_i^S$ where only saddle fixed points of (14) can lie.

Fixed points (u, v) whose images under ϕ lie above the right branch of κ are repellors. No (G_1, G_2) can lie in that region, if $C, A > 1/4$, that is, if $\delta(\alpha - 4) < \beta\gamma$ and $\alpha(\delta - 4) < \beta\gamma$, which is equivalent to $max\{\alpha(\delta - 4), \delta(\alpha - 4)\} < \beta\gamma$. □

The condition $max\{\alpha(\delta - 4), \delta(\alpha - 4)\} < \beta\gamma$ implies that when self-excitations of recurrent neurons are not significantly higher than their mutual inhibition, there are no repulsive fixed points of (14). As self-excitations α and δ grow, stable fixed points of (14) move closer towards $\{0, 1\}^2$. More precisely:

Corollary 1: *Same assumptions as in Theorem 2. All attractive fixed points of (14) lie in the ε-neighborhood of vertices of unit square, where*

$$\varepsilon = \sqrt{(0.5 - \Delta(\alpha))^2 + (0.5 - \Delta(\delta))^2}.$$

.

The tendency of attractive fixed points in discrete-time RNNs with exclusively self-exciting recurrent neurons to move towards saturation values as neural gain grows is also discussed in [21].

So far, we have confined the areas of the network state space $(0,1)^2$ where (under some assumptions on weights) fixed points of (14) of particular stability types can lie. In the following, it will be shown that those regions correspond to monotonicity intervals of functions defining fixed points of (14). The reasoning about the stability type of a fixed point can be based on the knowledge of where the functions intersect.

Recall that any fixed point (u_*, v_*) of (14) satisfies

$$(u_*, v_*) = (g(\alpha u_* + \beta v_*), g(\gamma u_* + \delta v_*)),$$

or equivalently, (v_*, v_*) lies on the intersection of two curves $v = f_{\alpha,\beta}(u), u = f_{\delta,\gamma}(v)$, where $f_{c_1,c_2} : (0,1) \to \Re$,

$$f_{c_1,c_2}(\ell) = -\frac{c_1}{c_2}\ell + \frac{1}{c_2} ln \frac{\ell}{1-\ell}. \tag{20}$$

$\lim_{\ell \to 0^+} f_{c_1,c_2}(\ell) = \infty$, $\lim_{\ell \to 1^-} f_{c_1,c_2}(\ell) = -\infty$[20]. f_{c_1,c_2} is convex and concave on $(0, 0.5)$ and $(0.5, 1)$ respectively. If $c_1 \le 4$, f_{c_1,c_2} is nonincreasing, otherwise it is decreasing on $(0, 0.5 - \Delta(c_1)) \cup (0.5 + \Delta(c_1), 1)$ and increasing on $(0.5 - \Delta(c_1), 0.5 + \Delta(c_1))$. Graph of $f_{c_1,c_2}(\ell)$ is presented in figure 13.

The "bended" graph of f_{c_1,c_2} for $c_1 > 4$ gives rise to a potentially complicated intersection pattern of $f_{\alpha,\beta}(u)$ and $f_{\delta,\gamma}(v)$. In the following, we shall consider only the case $c_1 > |c_2|$, since it is sufficient to explain some interesting features of training process observed in our experiments. Note that $c_1 > |c_2|$ means that for both neurons, the self-excitation is higher than the inhibition from the other neuron.

Lemma 2: *Assume $\alpha > 0, \beta < 0, \gamma < 0, \delta > 0$. If $\alpha \ge |\beta|$ and $\delta \ge |\gamma|$, then $f_{\alpha,\beta}(u)$ and $f_{\delta,\gamma}(v)$ do not intersect in* $(0, 0.5)^2$.

Proof: Assume that both $f_{\alpha,\beta}(u)$ and $f_{\delta,\gamma}(v)$ lie in $(0, 0.5)^2$, otherwise the result follows trivially. For $u \in (0, 0.5)$, both $(ln(u/(1-u)))/\beta$ and $-\alpha u/\beta$ are positive. It follows that in $(0, 0.5)^2$, $f_{\alpha,\beta}(u)$ lies above the line $v = \alpha u/|\beta|$. Similarly, in $(0, 0.5)^2$, $f_{\delta,\gamma}(v)$ lies above the line $u = \delta v/|\gamma|$. In terms of the co-ordinate system (u, v), this can be restated as follows: in $(0, 0.5)^2$, the graph of $f_{\alpha,\beta}$ lies above the line $v = \alpha u/|\beta|$ while the graph of $f_{\delta,\gamma}$ lies bellow the line $v = |\gamma| u/\delta$. Since $|\gamma|/\delta \le 1 \le \alpha/|\beta|$, $f_{\alpha,\beta}(u)$ and $f_{\delta,\gamma}(v)$ do not intersect in $(0, 0.5)^2$. □

[20] note that since α, δ and β, γ are assumed to be positive and negative respectively, we have $c_1 > 0$ and $c_2 < 0$

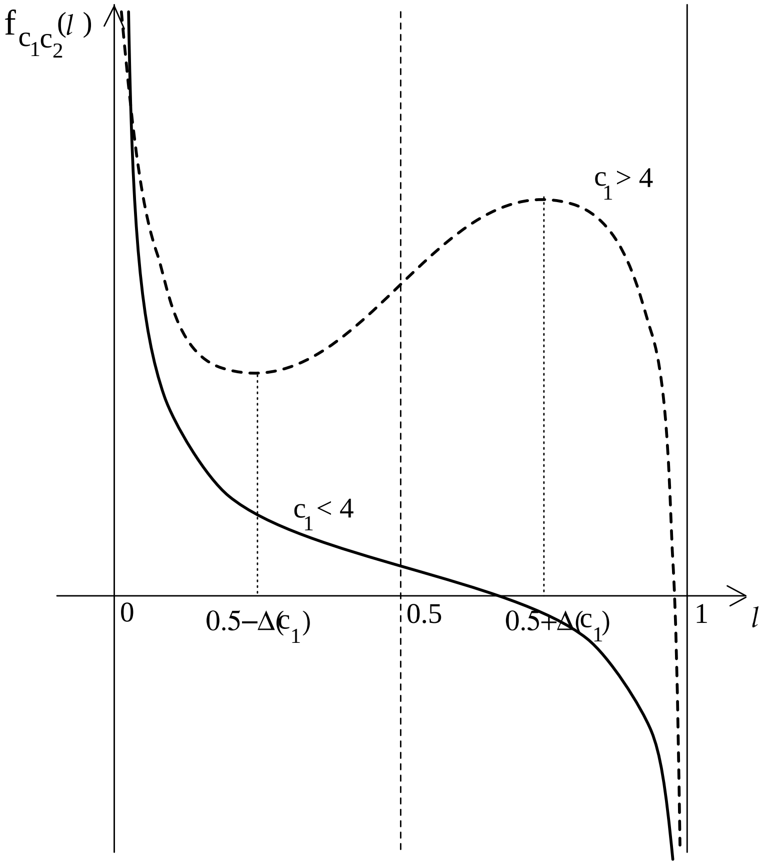

Figure 13:

The correspondence between regions $R^Q_{i,j}$, $i,j = 0,1$, $Q = A, S, R$, and the regions of monotonicity of $f_{\alpha,\beta}(u)$ and $f_{\delta,\gamma}(v)$ enables us to interpret training process as a process of "shaping" $f_{\alpha,\beta}$ and $f_{\delta,\gamma}$ so that the desired behaviour of (14), as prescribed by the training set, is achieved.

Denote the set $\{(u, f_{\alpha,\beta}(u)) |\ u \in (0, 0.5 - \Delta(\alpha))\}$ of points lying on the "first decreasing branch" of $f_{\alpha,\beta}(u)$ by $f^{0-}_{\alpha,\beta}$. Analogically, the set of points $\{(u, f_{\alpha,\beta}(u)) |\ u \in (0.5 + \Delta(\alpha), 1)\}$ in the "second decreasing branch" of $f_{\alpha,\beta}(u)$ is denoted by $f^{1-}_{\alpha,\beta}$. Finally, let $f^{+}_{\alpha,\beta}$ denote the set of points $\{(u, f_{\alpha,\beta}(u)) |\ u \in (0.5 - \Delta(\alpha), 0.5 + \Delta(\alpha))\}$ on the increasing part of $f_{\alpha,\beta}(u)$. Similarly, $f^{0-}_{\delta,\gamma}$, $f^{1-}_{\delta,\gamma}$ and $f^{+}_{\delta,\gamma}$ are used to denote the sets $\{(f_{\delta,\gamma}(v), v) |\ v \in (0, 0.5 - \Delta(\delta))\}$, $\{(f_{\delta,\gamma}(v), v) |\ v \in (0.5 + \Delta(\delta), 1)\}$ and $\{(f_{\delta,\gamma}(v), v) |\ v \in (0.5 - \Delta(\delta), 0.5 + \Delta(\delta))\}$ respectively. Using the Theorem 2 and Lemma 2 we state the following corollary:

Corollary 2: *Same assumptions as in Theorem 2. Attractive fixed points of (14) can lie only on the intersection of decreasing parts of $f_{\alpha,\beta}$ and $f_{\delta,\gamma}$. Whenever the increasing part of $f_{\alpha,\beta}$ intersects with a decreasing part of $f_{\delta,\gamma}$ (or vice-versa), it corresponds to a saddle point of (14). In particular, all attractive fixed points of (14) are from $f^{0-}_{\alpha,\beta} \cap f^{1-}_{\delta,\gamma}$, $f^{1-}_{\alpha,\beta} \cap f^{1-}_{\delta,\gamma}$ or $f^{1-}_{\alpha,\beta} \cap f^{0-}_{\delta,\gamma}$. Every point from $f^{+}_{\alpha,\beta} \cap f^{1-}_{\delta,\gamma}$ or $f^{1-}_{\alpha,\beta} \cap f^{+}_{\delta,\gamma}$ is a saddle point of (14).*

The usual scenario of creation of a new attractive fixed point of (14) is that typical of saddle-node bifurcation in which a pair attractive + saddle fixed point is created. Attractive fixed points disappear in a reverse manner: an attractive point coalesces with with a saddle and they are annihilated. This is illustrated in figure 14. $f_{\delta,\gamma}(v)$ shown as dashed curve intersects $f_{\alpha,\beta}(u)$ in three points. By increasing δ, $f_{\delta,\gamma}$ bends further (solid curve) and intersects with $f_{\alpha,\beta}$ in five points[21]. Saddle and attractive points are marked with squares and circles respectively. Note that as δ increases attractive fixed points move closer to vertices $\{0,1\}^2$.

A similar approach to determining the number and stability types of fixed points of the underlying dynamical systems in continuous–time recurrent neural networks can be found in [2].

[21] At the same time, $|\gamma|$ has to be also appropriately increased so as to compensate for the increase in δ so that the "bended" part of $f_{\delta,\gamma}$ does not move radically to higher values of u.

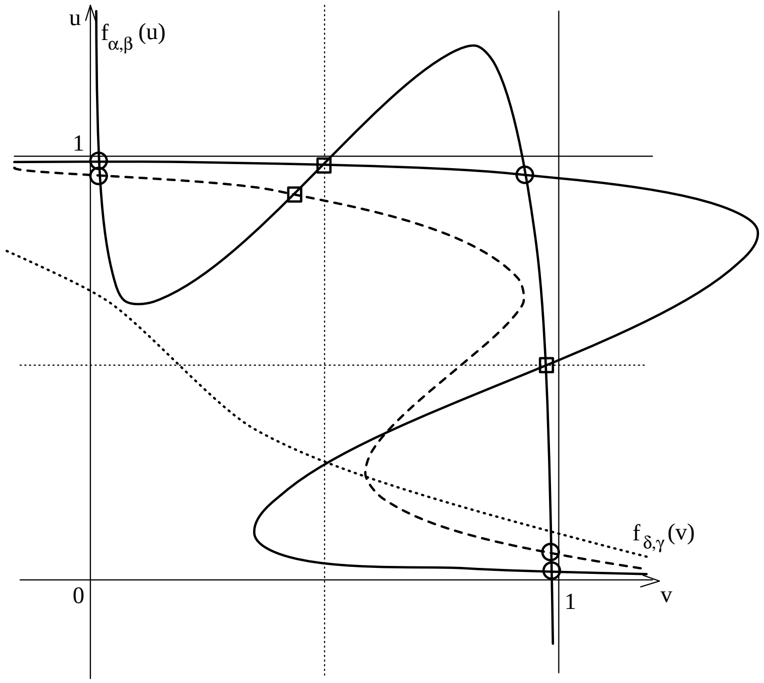

Figure 14: Geometrical illustration of saddle-node bifurcation in RNN with two state neurons.

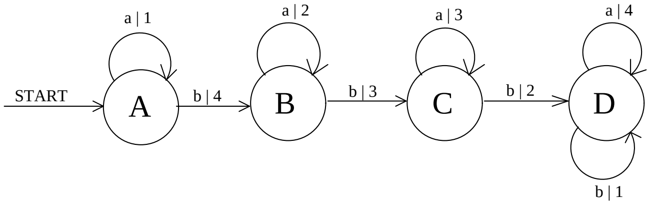

Figure 15: FSM $\mathcal{M}$ with four a-loops and "transition" input symbol b.

8 Experiments – Learning loops of FSM

A RNN with two state neurons was trained with the FSM $\mathcal{M}$ presented in figure 15. In each of its four states there is an a-loop. Input symbol b causes subsequent transitions between states up to the "trap" state D. Training set representing $\mathcal{M}$ was constructed as follows: Transitions to states B, C and D from the initial state A are represented by one, two and three consecutive b's respectively. Apart from transition, each a-loop is represented by strings of consecutive a's up to length 5. b-loop in the state D is represented by a string of 5 consecutive b's. To each input string w, its corresponding output string $\lambda^+(A, w)$ is determined.

During the training, after each epoch, attractive sets of τ_a were numerically detected. The evolution of position and number of attractive fixed point(s) of τ_a in $(0,1)^2$ can be seen in figure 16.

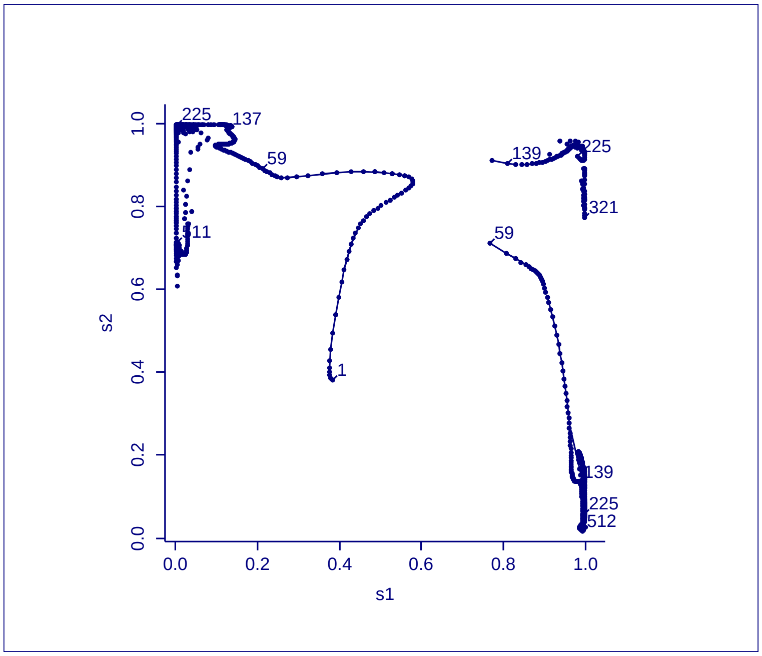

Figure 16: Evolution of position of attractive sets of τ_a during RNN training on FSM $\mathcal{M}$ (two state neurons).

Near the points the corresponding epoch numbers are shown. At the beginning, there is only one fixed point of τ_a. A bifurcation during the 59th epoch produces two attractive fixed points. Since the 138th epoch till the 321st epoch there are three attractive fixed points and two saddle points of τ_a. These are determined by the intersection of the corresponding lines f_{α_a,β_a} and f_{δ_a,γ_a}, where $\alpha_a, \beta_a, \gamma_a$ and δ_a are coefficients of the map τ_a as in (14). The episode of existence of the attractive fixed point $f^{1-}_{\alpha_a,\beta_a} \cap f^{1-}_{\delta_a,\gamma_a}$ begins when f_{α_a,β_a} is "bended" enough so that $f^{1-}_{\delta_a,\gamma_a}$ intersects with both increasing and decreasing parts $f^{+}_{\alpha_a,\beta_a}$ and $f^{1-}_{\alpha_a,\beta_a}$ respectively. At the same time, in order for the intersection $f^{1-}_{\alpha_a,\beta_a} \cap f^{+}_{\delta_a,\gamma_a}$ to exist, f_{δ_a,γ_a} needs also to be sufficiently "bended" (figure 17). The degree to which f_{α_a,β_a} and f_{δ_a,γ_a} are "bended" is primarily controlled by α_a and δ_a respectively, while the vertical positions of bended parts are mainly determined by respectively β_a and γ_a. During the 322nd epoch, the attractive fixed point $f^{1-}_{\alpha_a,\beta_a} \cap f^{1-}_{\delta_a,\gamma_a}$ together with saddle point $f^{1-}_{\alpha_a,\beta_a} \cap f^{+}_{\delta_a,\gamma_a}$ disappear because the increase in $|\gamma_a|$ pushes the "bended" part of f_{δ_a,γ_a} inside the state space $(0,1)^2$ (figure 18).

The training error was 0.08, yet the only attractive sets of τ_a that were detected were two attractive fixed points S_A and S_D near vertices $(0,1)$ and $(1,0)$ corresponding to a-loops in states

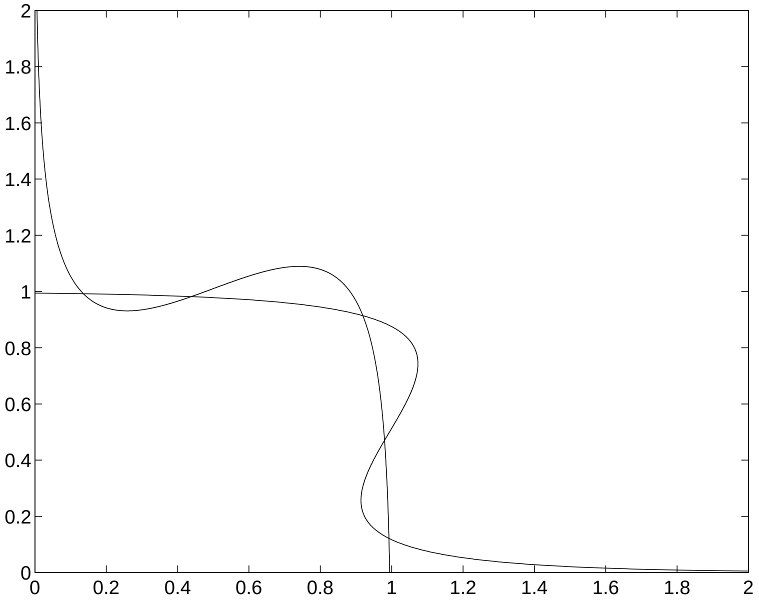

Figure 17: $f_{\alpha_a,\beta_a}(u)$ and $f_{\delta_a,\gamma_a}(v)$ after 150th training epoch. Coefficients of the map τ_a are $\alpha_a = 5.21, \beta_a = -2.58, \gamma_a = -2.63, \delta_a = 5.23$.

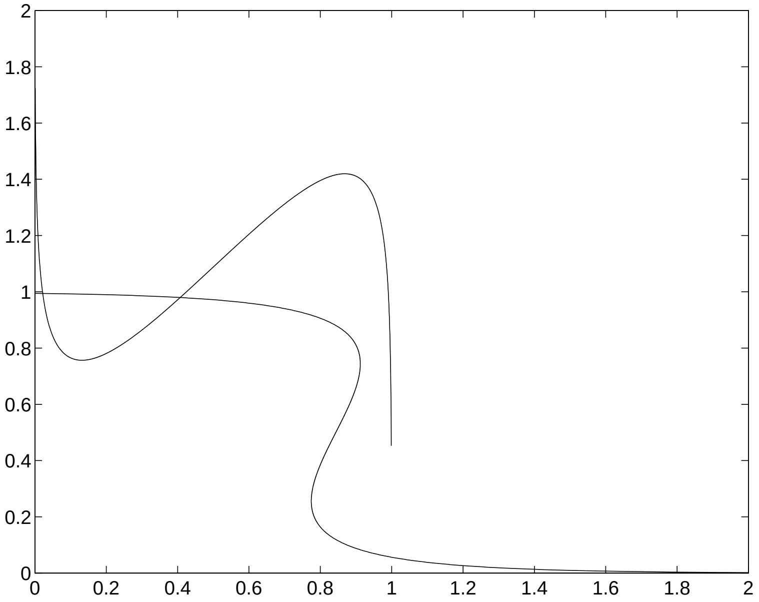

Figure 18: $f_{\alpha_a,\beta_a}(u)$ and $f_{\delta_a,\gamma_a}(v)$ after 1000th training epoch. Coefficients of the map τ_a are $\alpha_a = 8.61, \beta_a = -3.96, \gamma_a = -3.08, \delta_a = 5.17$.

A and D respectively. Starting in a small neighborhood of S_A and S_D, upon repeated presentation of input a, the decoded network outputs are 1 and 4 with trajectories of τ_a approaching S_A and S_D respectively. There is no stable representation of the a-loops in states B and C, i.e. there are no positively invariant sets of τ_a leading to the network output 2 and 3 respectively when input a is presented to the network.

However, the net is able to simulate the training set perfectly. It follows that after it is reset[22] and presented with b, when five consecutive a's arrive, the decoded output will be five consecutive 2's. Hence, the network must have developed a mechanism for acting as if the a-loops in B and C were represented in a stable manner, at least for strings having no more than five consecutive a's. It turns out that the underlying mechanism for pretending that there are stable representations of a-loops for short input strings involves a behaviour of trajectories starting "near" the stable manifold W^s of the saddle fixed point S_S lying "between" attractive points S_A and S_D, with W^s constituting the border of regions of attraction of S_A and S_D.

Consider a point S "near" W^s. Due to the continuity of τ_a, the orbit of S under τ_a first moves towards S_S along W^s and then away from S_S along a branch of the unstable manifold W^u of S_S gradually approaching one of the attractive points S_A, S_D. To which of the two points the trajectory actually converges is determined by the "side" of W^s on which the initial point S lies. Assume that the trajectory of S converges to S_A. If we slightly displace S into S' on "the other side" of the curve W^s, trajectories trajectories of S and S' move towards S_S close to each other, but as they approach S_S, the trajectory of S' follows the other branch of W^u towards S_D (see figure 19). As we move starting point S towards S_A and S_D, the trajectories less and less follow the pattern described above, move towards S_A and S_D in a straightforward manner[23] and approach a vicinity of S_A and S_D respectively much faster than trajectories starting "near" W^s. Hence, the network is able to "cheat" by pretending stable behaviour as described by the a-loop in the state B because it takes advantage of different convergence rates of orbits starting near W^s and S_D. The decoded output of the net with input a and a state near S_D is 4 (region $\mathcal{D}$), while for states involving first several steps in trajectories starting near W^s, the output is 2 (region $\mathcal{B}$). Analogical statement can

[22] with (possibly repeated) presentation of "reset" input #

[23] Due to the coefficients of τ_a, eigenvalues of its Jacobian in every point from $(0,1)^2$ are real thus implying an absence of rotation in neighborhoods of fixed points.

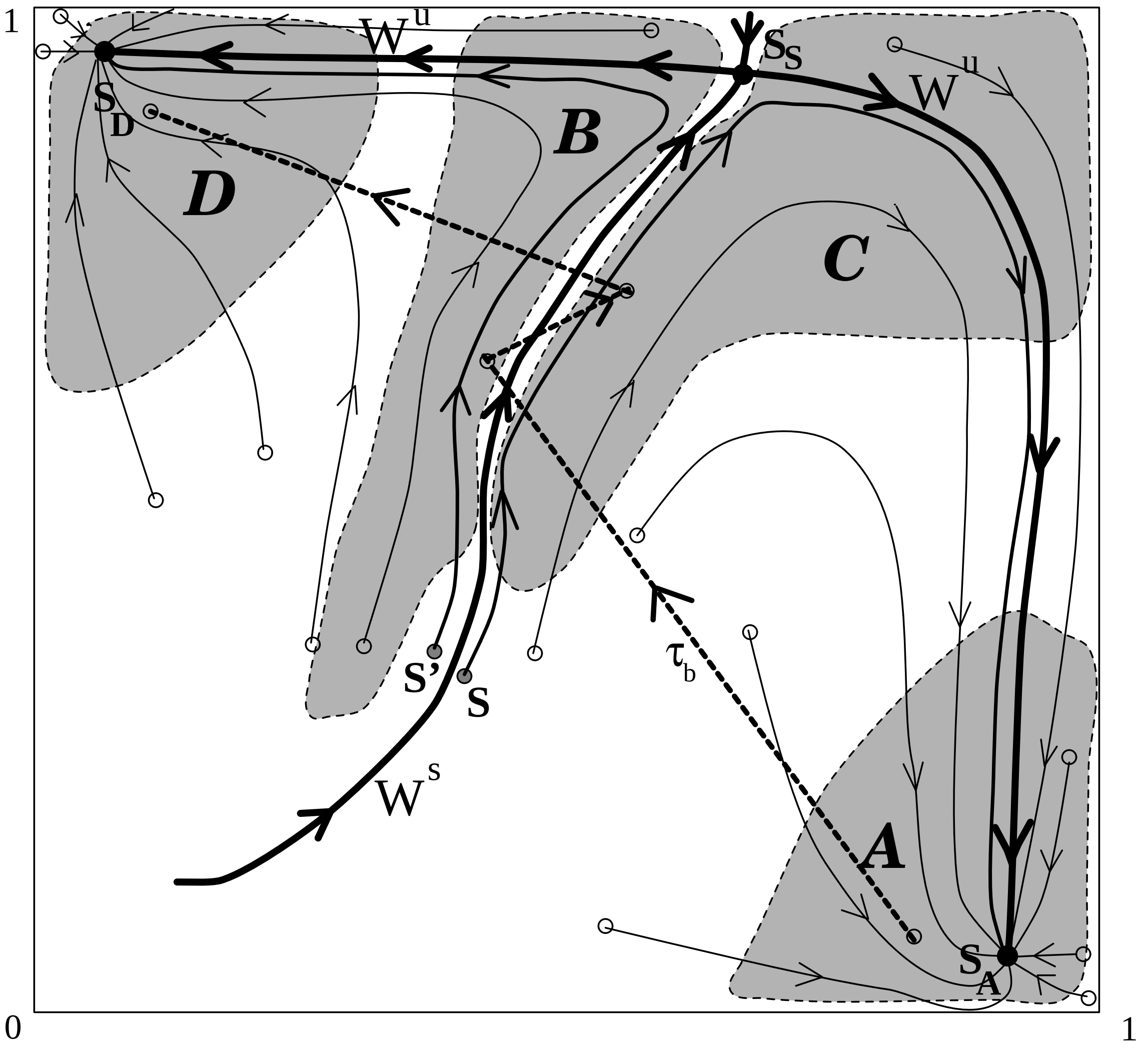

Figure 19: Illustration of a mechanism that enables RNN to "pretend" stable representation of loops in $\mathcal{M}$ for short input strings.

be made about trajectories starting near S_A and W^s, and regions $\mathcal{A}$ and $\mathcal{C}$ respectively. Most of the time towards the end of learning session was spent on learning the output function $\nu_a(S) = \nu(S, a)$ in closely neighboring regions of $\mathcal{B}$ and $\mathcal{C}$ so that the outputs for states from $\mathcal{B}$ and $\mathcal{C}$ are 2 and 3 respectively (see figures 20, 21). The map $\tau_\#$ associated with the "reset" input symbol # has one attractive fixed point in the region $\mathcal{A}$. Under the "reset" map $\tau_\#$, trajectories of network states $S \in (0,1)^2$ quickly approach region $\mathcal{A}$ thus preparing ground for processing of a new input word.

The key role, however, is played by the transfer function τ_b. It simulates transition between states with a-loops in $\mathcal{M}$. Starting in $S \in \mathcal{A}$, $\tau_b(S) \in \mathcal{B}$ and $\tau_b^2(S) \in \mathcal{C}$ lie near W^s and the behaviour of τ_a in $\mathcal{B}$ and $\mathcal{C}$ appears to be stable for several iterations. Upon repeated presentation

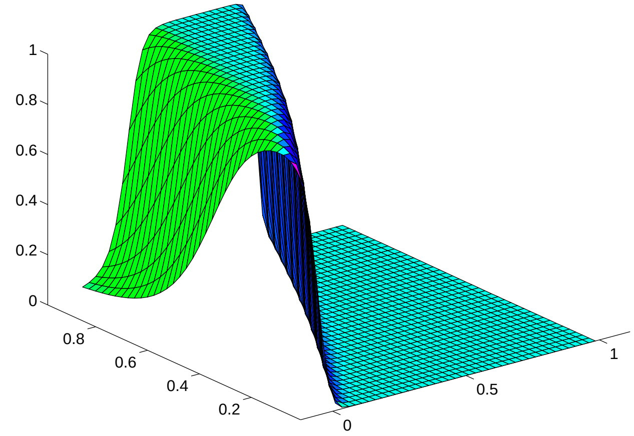

Figure 20: The map $(\nu_a)_2$ representing the output of the second output neuron that corresponds to the output symbol 2. Note the sharp activity change along border of regions of attraction of S_A and S_D.

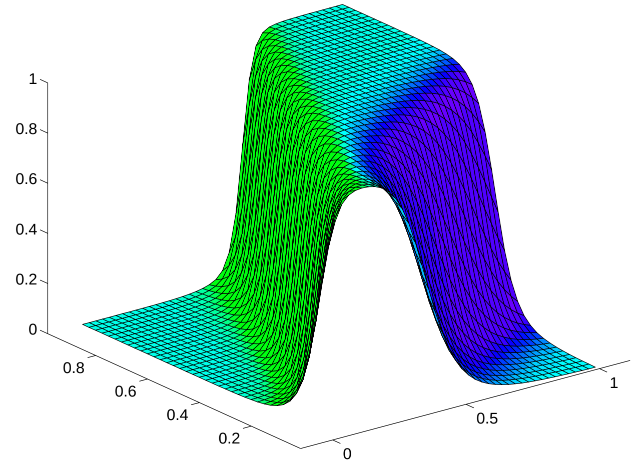

Figure 21: The map $(\nu_a)_3$ representing the output of the third output neuron that corresponds to the output symbol 3. A sharp activity change along border of regions of attraction of S_A and S_D is clearly visible.

of a, $\tau_b^3(S) \in \mathcal{D}$ converges to S_D with network output 4.

The delicate role of τ_b responsible for transitions $\mathcal{A} \to \mathcal{B} \to \mathcal{C} \to \mathcal{D}$ with jumping on the "appropriate" sides of W^s while staying close to W^s, together with different convergence rates of orbits under τ_a starting close to W^s and near S_A, S_D are principal tools enabling the net to behave nicely for testing strings of smaller length, although it generalizes poorly on strings with many consecutive a's after b or bb. In particular, the outputs of the net for input strings ba^n and bba^m are consistent with training set for $n = 8$ and $m = 10$. As further a's keep coming, trajectories of τ_a move away from $\mathcal{B}$ and $\mathcal{C}$ towards S_D and S_A respectively.

To visualize the process of state degradation upon repeated presentation of input a a *state degradation diagram for input* a is constructed as follows (M_a denotes the set of states of $\mathcal{M}$ in which there is an a-loop):

- Construct a finite vocabulary Γ of short distinguishing words for M_a, such that Γ does not contain a word $ua^iv, i \geq 2$, where u is leading to a state of $\mathcal{M}$ in which there is an a-loop. With each state q of M_a associate a minimal input word m_q leading to q.
- For each $i \in \{1, 2, ..., N_{max}\}$
    - For each $w \in \Gamma$
        - For each state $q \in M_a$
            - present the reset network with $m_q a^i$ and then
            - present the network with w and check whether the net output equals $\lambda^+(q, w)$. If not, check whether there is a state p of $\mathcal{M}$ such that the network output equals $\lambda^+(p, w)$ – if so, draw an arrow in a diagram from q to p.

State degradation diagram for input a is presented in figure 22. Note that when only short input strings are presented to the network, and quantization of network state space individually captures regions $\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}$ a correct state transition diagram can be obtained, even though, on longer input strings the net generalizes poorly.

When the network with three state neurons was trained with the FSM $\mathcal{M}$, it generalized correctly over the training set by forming four attractive fixed points of τ_a corresponding to loops in states A, B, C, D of $\mathcal{M}$. The training process looked at from the point of view of asymptotic

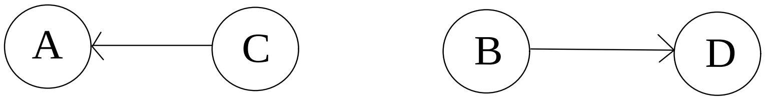

Figure 22: State degradation diagram for input a. $N_{max} = 100$.

behaviour of τ_a is illustrated in figure 23. Horizontal axis correspond to time (in epochs), network state space $(0,1)^3$ is orthogonally projected into 2-dimensional space of activations of a couple of state neurons. Bifurcations leading to formation of new attractive fixed points appeared during the 53rd,115th and the 121st epoch. If the network is able to exactly mimic the FSM $\mathcal{M}$ the state degradation diagram for each input symbol has no arrows.

As another example, Consider a FSM $\mathcal{M}$ in figure 24. It is a FSM taken from the database of the *International Symposium on Circuits and Systems* (Portland, Oregon, 1989) [4]. In each of its 7 states there is an a-loop with output 0 except for a-loops in states 4 and 7. The training set consists of 3500 training strings[24] of input string length 3–35 and is ordered according to their length starting with the shortest ones. The machine $\mathcal{M}$ is hard to learn because the training set is very sparse in output symbols other than 0. Training process is disrupted by a tendency to find trivial solution represented by the automaton with only one state and loops for every input symbol with the output 0. An example of a part of the training set is given in table 1.

After 53 training epochs RNN with 6 state neurons is able to perform well on short test strings (training error was 0.06). Generalization on long test strings was found to be poor. Part of the problem was unstable network representation of a-loops in $\mathcal{M}$. The state degradation diagram for input a can be seen in figure 25. a-loops in states 4,6 and 7 are "well represented" by fixed points S_4, S_6 and S_7 respectively in that when starting in a small neighborhood of $S_q,\ q = 4,6,7$, the resulting output sequences of RNN for input words $a^iw,\ w \in \Gamma,\ i \geq 0$ equal $\lambda^+(q, a^iw)$. This is not true of a-loops in states 1,2,3 and 5. When the net is reset and presented with $m_q,\ q = 1,2,3,5$, for $i > N_q$ it does not emulate $\lambda^+(q, a^iw),\ w \in \Gamma$. Sates 5 and 3 degradate to states 1 and 2 respectively. In particular, $N_5 = 8$ and $N_3 = 5$. Both states 1 and 2 degradate to attractive fixed point S_0 with $N_1 = 27$ and $N_2 = 40$. The network state S_0 does not represent any state of $\mathcal{M}$ even for short input strings. $S_j,\ j = 0,4,6,7,$ are the only attractive sets of τ_a that were detected. There are trajectories of τ_a starting near border of regions of attraction of S_0 and some

[24] input word $w \rightarrow$ corresponding output word $\lambda^+(q_0, w)$

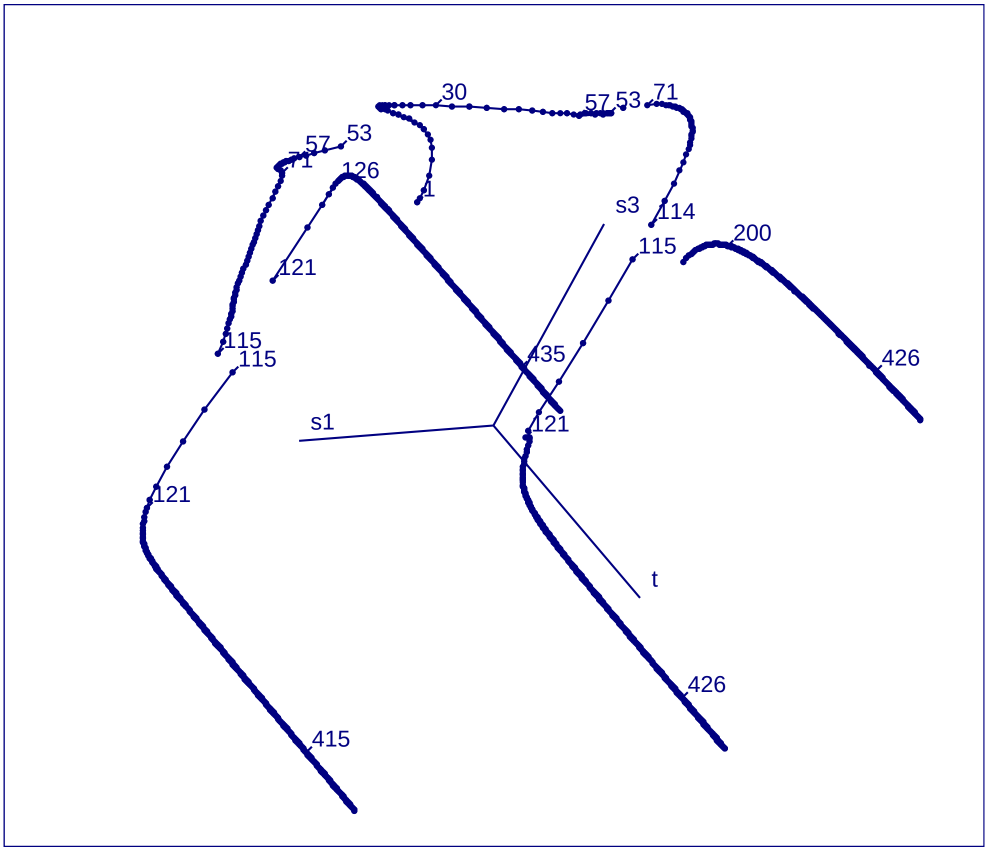

Figure 23: Evolution of position of attractive sets of τ_a during RNN training on FSM $\mathcal{M}$ (three state neurons).

```
.

.

.

dddeadfdaeaafaaadddaddadfeeedeaeee# --> 0000000000000000000000000000000002x

affedfeefaedeededfdefddaafeeeeeadd# --> 0000000000000000000000000000022200x

dffdadedfadaddffeeafeafdffdffefaad# --> 0000000000000000000000000000000000x

fdaadaafddafafdadfdffdeaffaaefeade# --> 0000000000000000000000000000000000x

ddfaddadfaaddddeafdafdfaeedaedeeda# --> 0000000000000000000000000000000000x

defadedefdeffdefdafdaaadeaeddaaefd# --> 0000000000000000000000000000000000x

ddfedaaffdedeaeadeefdfefaadadeaaff# --> 0000000000000000000000000000000000x

aafaaeefafeaffeeefeafaefeeadaefafa# --> 0000000000000000000000000000000000x

dddeeafffafeaadaddfdffadfeafdddefd# --> 0000000001100000000000000000000000x

fdaaddaadadffefaeadddfeddeafdddaea# --> 0000000000000000000000000000000000x

dedaddadaafeaaddaafaaefaefdeeffafe# --> 0000000000000000000000000000000000x

ddaeeafddfaaffffaeeefeadaefdfedfee# --> 0000000000000011100000000000000000x

dddedeeafdfddfaeeaddafdfafadedfaaf# --> 0000000000000000000000000000000000x

.

.

.
```

Table 1: A part of the training set characterizing the FSM $\mathcal{M}$. Output strings are sparse in output symbols other than 0.

other attractive fixed point of τ_a that pass through the region assuming the role of state 5 of $\mathcal{M}$ for short input strings. Then, further towards S_0, they pass through the region of network states that for short input strings seem to be equivalent to the state 1 of $\mathcal{M}$, finally making their way to a close neighborhood of S_0 and converge to it. A similar statement can be made about states 3 and 2 of $\mathcal{M}$.

9 Discussion

Two views on the relationship between a RNN and a FSM $\mathcal{M}$ such that the RNN exactly mimics $\mathcal{M}$ were presented. First, the network was treated as a state machine. The notion of regions of equivalent network states that are also equivalent to a state of $\mathcal{M}$ link the first approach with the second, dynamical systems' approach to the RNN.

Our experiments suggest that the most usual stable RNN $\mathcal{N}$ representations of loops and cycles in $\mathcal{M}$ can be described as follows: An x-loop in a state q of $\mathcal{M}$ induces an attractive fixed point of τ_x inside $(q)_{\mathcal{N}}$, and an x-cycle $\{q_1, ..., q_m\}$ of $\mathcal{M}$ induces an attractive periodic orbit of period m of τ_x periodically visiting $(q_1)_{\mathcal{N}}, ..., (q_m)_{\mathcal{N}}$.

The present paper provides us with the opportunity to look at the learning process from the point of view of bifurcation analysis. If the network is supposed to operate as a FSM, its state space must have multiple attractor basins to store distinct internal states. The network solves the task of FSM simulation by location of point and periodic attractors and the shaping of their respective basins of attraction [9]. Before training, the connection weights are set to small random values and as a consequence, the network has only one attractor basin. This implies that the network must undergo several bifurcations [13]. This can have an undesirable effect on the training process, since the gradient descent learning may get into trouble. At bifurcations points, the output of a network can change discontinuously with the change of parameters and therefore convergence of gradient descent algorithms is not guaranteed [14].

In the following a possible application of these ideas to the problem of determination of the complexity of language recognition by neural networks will be discussed briefly.

Any FSM with binary output alphabet $\{0, 1\}$ can function as a recognizer of a regular language. A word over the input alphabet belongs to the language only if the output symbol after presentation

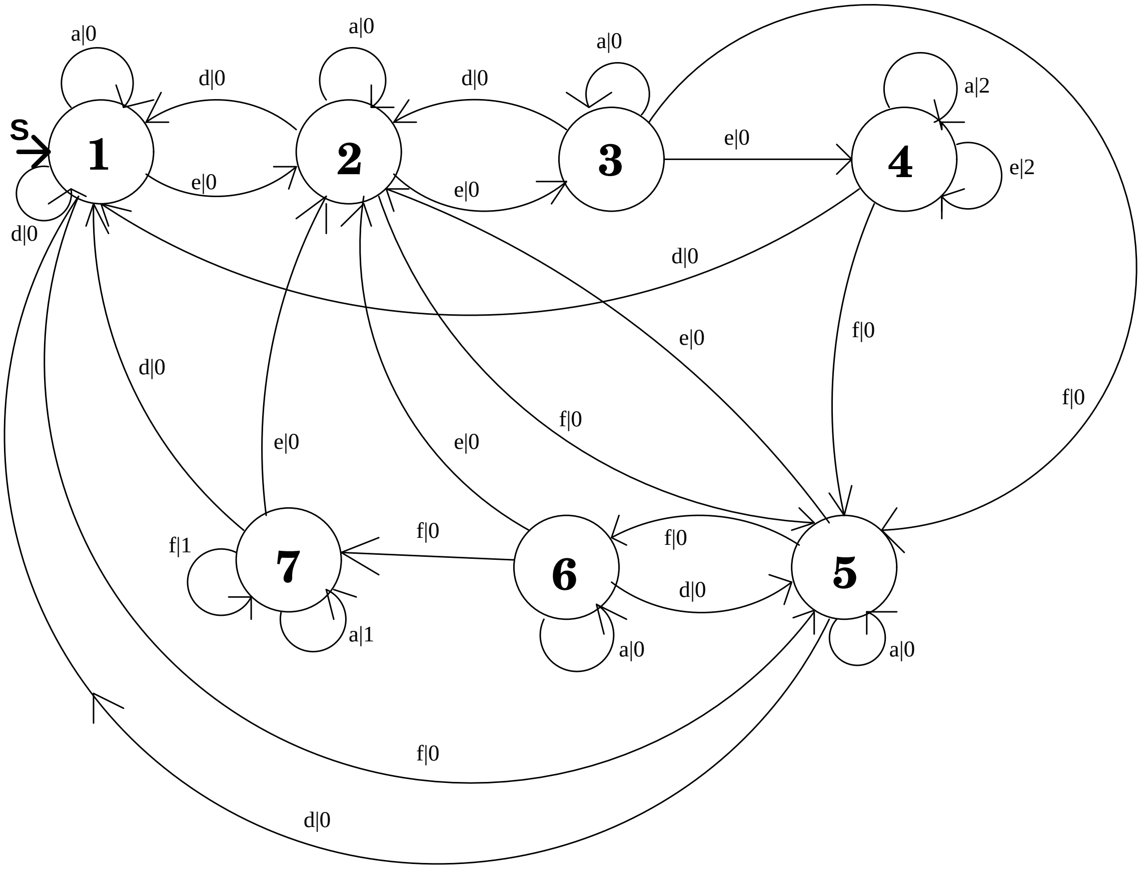

Figure 24: FSM $\mathcal{M}$ taken from the database of the International Symposium on Circuits and Systems (Portland, Oregon, 1989). $\mathcal{M}$ is the reduced form of a machine defined in the file `bbara.kiss2`. Inputs $--01, --10$ and $--00$ are represented as the input symbol a since, in every state, they initiate the same transition with the same output. Inputs $0011, -111$ and 1011 are represented as input symbols d, e and f respectively. Outputs $00, 01$ and 10 are coded as output symbols $0, 1$ and 2 respectively.

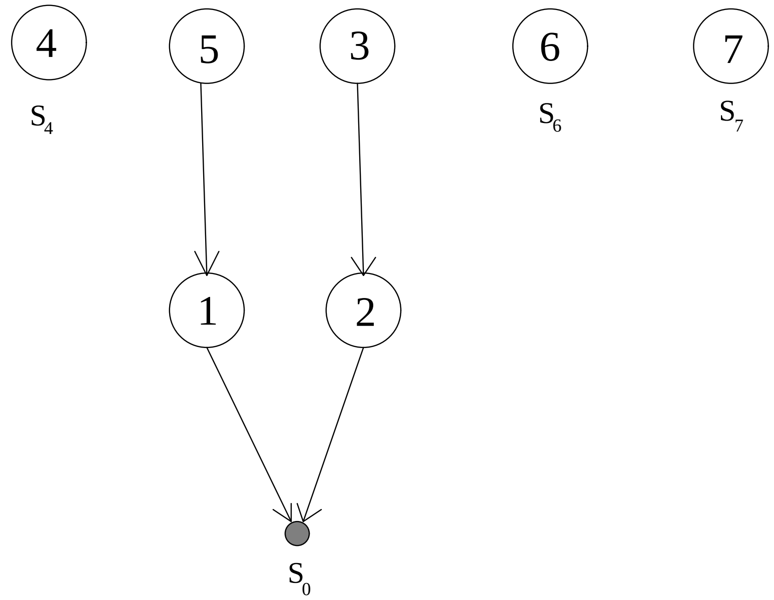

Figure 25: State degradation diagram for input a extended with network state S_0 not representing any state of $\mathcal{M}$. $S_0 = (0.89, 0.01, 0.55, 0.95, 0.99, 0.92), S_4 = (0.16, 0.98, 0.02, 0.87, 0.04, 0.92), S_6 = (0.98, 0.03, 0.97, 0.09, 0.99, 0.87), S_7 = (0.94, 0.98, 0.95, 0.01, 0.05, 0.15)$. $N_{max} = 100$.

of word's last symbol is 1. Hence, the network output is used to decide whether a word does belong to the language, or not. One of the most promising neural acceptors of regular languages [32] is the second-order RNN introduced by Giles *et al.* [17]. However, the practical aspects of the acceptance issue are still unclear [33]. The difficulty of acceptance of a given language by a neural network (the neural complexity of the language) can be quantified by the minimal number of neurons needed to recognize the language. In the context of mealy machines and threshold networks a similar problem was attacked by Alon *et al.* [1] and Horne and Hush [23]. An attempt to predict the minimal second-order RNN size so that the network can learn to accept a given regular language is presented in [33]. The predicted numbers of neurons were shown to correlate well with the experimental findings.

Essentially, a good starting point for the estimation of neural complexity of a given regular language is the representation of the language with the reduced recognizer. The most usual, very rough, approach to the neural complexity estimation takes into account only the number of states of such a recognizer [33]. What plays a principal role in making the internal structure of a regular language rich is

- the number of input symbols of the recognizer
- the number of loops associated with each input symbol
- the number and corresponding lengths of cycles associated with each input symbol
- the relationship among loops and/or cycles (i.e. a x_1-cycle is passing through a state q in which there exists a x_2-loop, etc...).

In every recognizer of a regular language, for each input symbol there exists at least one loop or a cycle. During the training process, the weights of a network are modified so that the corresponding attractive sets evolve in dynamical systems defined by the iterative maps τ_x. A hint for a lower bound on the minimal number of neurons can be obtained by exploring the possibilities of the existence of attractive points and/or periodic orbits that are to be induced during the training process. The expected relationship among their basins of attraction has to be taken into account at the same time [5].

As an example consider the FSMs $\mathcal{M}_1$ and $\mathcal{M}_2$ presented in figures 26, and 27 respectively. Apparently, the the internal structure of a regular language accepted by $\mathcal{M}_2$ is "more complex" than that of accepted by $\mathcal{M}_1$. In the latter case, only one attractive fixed point of τ_a is sufficient to represent the a-loop in the state E. The same applies to the b-loop in E, and the map τ_b. In the former case, an attractive periodic orbit of period four of the map τ_a, and four attractive points of the map τ_b have to be induced. Even though the FSM $\mathcal{M}_2$ has only four states, the RNN needed four state neurons to accomplish a successful learning. On the other hand, two state neurons were sufficient for the RNN to learn the FSM $\mathcal{M}_1$.

A mechanism underlying generalization loss on longer input strings due to unstable representation of loops in a FSM to be learned was investigated. It was shown that even in such cases a correct state transition diagram of the FSM can potentially be extracted even though the network performs badly on longer input strings (as reported by Giles *et al.* [17]). The state degradation diagram for an input symbol x illustrates how regions of network state space, initially acting as if they assumed the role of states of the FSM in which there is an x-loop, gradually degradate upon repeated presentation of x. The degradation may lead to a network state not representing any state of the FSM even for short input strings.

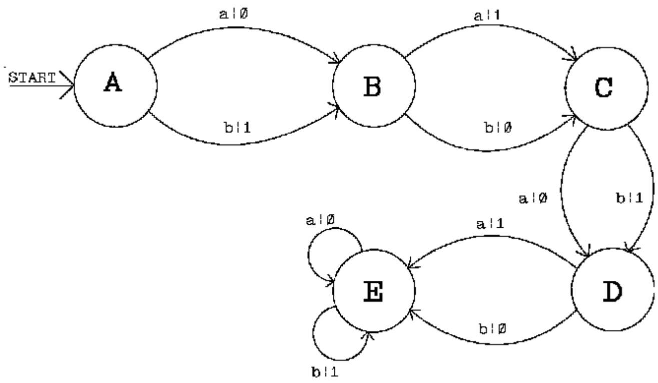

Figure 26: Acceptor of the language $L = L_1 \cup L_2,\ L_1 = \{a,b\}^n b,\ n \in \{0,2,4,5,6,...\},\ L_2 = \{a,b\}^m a,\ m \in \{1,3\}$.

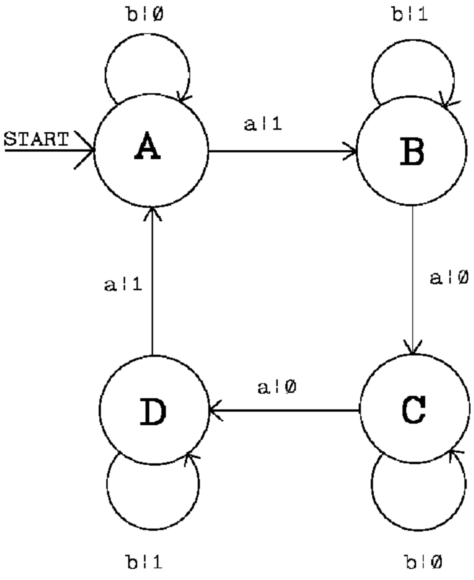

Figure 27: Acceptor of the language $L = L_3^+$, where $L_3 = b^*ab^* \cup (b^*a)^3b^+ \cup (b^*a)^4$.

Zeng *et al.* [39] and Das and Mozer [11] view the RNN state space quantization as an integral part of the learning process in which the network is trained to mimic a finite state machine. In particular, in [39] state units' activation pattern is mapped at each time step to the nearest corner of a hypercube as if state neurons had a hard threshold activation function. Das and Mozer [11] used a "soft" version of the gaussian mixture model[25] in a supervised mode as a clustering tool. The mixture model parameters were adjusted so as to minimize the overall performance error of the whole system (recurrent network + clustering tool). Both Zeng *et al.*, and Das and Mozer report better assymptotical behaviour for long, unseen test input strings. It would be interesting to investigate such approaches to training RNN on finite state problems as a form of "dynamical self-reinforcement" learning encouraging bifurcations to attractive fixed points and periodic orbits of the underlying dynamical systems.

Acknowledgments

Thanks to Mária Markošová, Pavol Brunovský and Phil Holmes for useful discussions on dynamical systems. Work of Mike Casey and Randall Beer was of great contribution in preparing this report.

References

[1] N. Alon, A.K. Dewdney, and T.J. Ott. Efficient simulation of finite automata by neural nets. *Journal of the Association of Computing Machinery*, 38(2):495–514, 1991.

[2] R.D. Beer. On the dynamics of small continuous–time recurrent networks. Technical Report CES–94–18, Case Western Reserve University, Cleveland, OH, 1994.

[3] E.K. Blum and X. Wang. Stability of fixed points and periodic orbits and bifurcations in analog neural networks. *Neural Networks*, (5):577–587, 1992.

[4] F. Brglez, D. Bryan, and K. Kozminski. Combinational Profiles of Sequential Benchmark Circuits. In *Proceedings of the International Symposium on Circuits and Systems*, Portland, Oregon, May 1989.

[25] Instead of the center with greatest posterior probability given a pattern of state units' activation, a linear combination of centers is used, where each center is weighted by its posterior probability given current network state.

[5] M.P. Casey. Computation dynamics in discrete-time recurrent neural networks. In *Proceedings of the Annual Research Symposium*, volume 3, pages 78–95, UCSD, La Jolla, CA, 1993. Institute for Neural Computation.

[6] M.P. Casey. *Computation in Discrete-Time Dynamical Systems.* PhD thesis, University of California, San Diego, Department of Mathematics, March 1995.

[7] M.P. Casey. Relaxing the symmetric weight condition for convergent dynamics in discrete-time recurrent networks. Technical Report INC-9504, Institute for Neural Computation, University of California, San Diego, 9500 Gilman Drive, La Jolla, CA 92093-0112, 1995.

[8] A. Cleeremans, D. Servan-Schreiber, and J.L. McClelland. Finite state automata and simple recurrent networks. *Neural Computation*, 1(3):372–381, 1989.

[9] F. Cummins. Representation of temporal patterns in recurrent networks. *Submitted to the 15th Annual Conference of the Cognitive Science Society*, 1993.

[10] S. Das, C.L. Giles, and G.Z. Sun. Learning context-free grammars: Capabilities and limitations of a recurrent neural network with an external stack memory. In *Proceedings of The Fourteenth Annual Conference of Cognitive Science Society.* Indiana University, 1992.

[11] S. Das and M.C. Mozer. A unified gradient–descent/clustering architecture for finite state machine induction. In J.D. Cowen, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 19–26. Morgan Kaufmann, 1994.

[12] R.L. Devaney. *An Introduction to Chaotic Dynamical Systems.* Benjamin/Cummings Publishing Company, Inc., Menlo Park, California, 1986.

[13] K. Doya. Bifurcations in the learning of recurrent neural networks. In *Proc. of 1992 IEEE Int. Symposium on Circuits and Systems*, pages 2777–2780, 1992.

[14] K. Doya. Bifurcations of recurrent neural networks in gradient descent learning. *Submitted to IEEE Transactions on Neural Networks*, 1993.

[15] J.L. Elman. Finding structure in time. *Cognitive Science*, 14:179–211, 1990.

[16] M. Garzon and S. Franklin. Global dynamics in neural networks. *Complex Systems*, (3):29–36, 1989.

[17] C.L. Giles, C.B. Miller, D. Chen, H.H. Chen, G.Z. Sun, and Y.C. Lee. Learning and extracting finite state automata with second–order recurrent neural networks. *Neural Computation*, 4(3):393–405, 1992.

[18] C.L. Giles, C.B. Miller, D. Chen, G.Z. Sun, H.H. Chen, and Y.C. Lee. Extracting and learning an unknown grammar with recurrent neural networks. In J.E. Moody, S.J. Hanson, and R.P. Lippmann, editors, *Advances in Neural Information Processing Systems 4*, pages 317–324, 1992.

[19] J. Guckenheimer and P. Holmes. *Nonlinear Oscilations, Dynamical Systems, and Bifurcations of Vector Fields*. Springer-Verlag, 1982.

[20] M.W. Hirsch. Convergent activation dynamics in continuous time networks. *Neural Networks*, 2(5):331–349, 1989.

[21] M.W. Hirsch. Saturation at high gain in discrete time recurrent networks. *Neural Networks*, 7(3):449–453, 1994.

[22] J.J. Hopfield. Neurons with a graded response have collective computational properties like those of two–state neurons. *Proceedings of the National Academy of Science USA*, 81:3088–3092, May 1984.

[23] B.G. Horne and D.R. Hush. Bounds on the complexity of recurrent neural network implementations of finite state machines. In J.D. Cowen, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 359–366. Morgan Kaufmann, 1994. Also submitted to Neural Networks.

[24] S. Hui and S.H. Zak. Dynamical analysis of the brain-state-in-a-box neural models. *IEEE Transactions on Neural Networks*, (1):86–94, 1992.

[25] L. Jin, P.N. Nikiforuk, and M.M. Gupta. Absolute stability conditions for discrete-time recurrent neural networks. *IEEE Transactions on Neural Networks*, (6):954–963, 1994.

[26] M.I. Jordan. Attractor dynamics and parallelism in a connectionist sequential machine. In *Proceedings of the Eighth Conference of the Cognitive Science Society*, pages 531–546. Erlbaum, 1986.

[27] L.K. Li. Fixed point analysis for discrete-time recurrent neural networks. In *Proceedings of IJCNN*, volume 4, pages 134–139, Baltimore, USA, 1992.

[28] P. Manolios and R. Fanelli. First order recurrent neural networks and deterministic finite state automata. *Neural Computation*, 6(6):1155–1173, 1994.

[29] R.C. Minnick. Linear–input logic. *IRE Transactions on Electronic Computers*, EC–13:6–16, 1961.

[30] P. Tiňo, I.E. Jelly, and V. Vojtek. Non-standard topologies of neuron field in self-organizing feature maps. In *Proceedings of the AIICSR'94 conference, Slovakia*, pages 391–396. World Scientific Publishing Company, 1994.

[31] P. Tiňo and J. Sajda. Learning and extracting initial mealy machines with a modular neural network model. *Neural Computation*, 7(4), 1995.

[32] M.W. Shields. *An Introduction to Automata Theory.* Blackwell Scientific Publications, London, UK, 1987.

[33] H.T. Siegelmann, E.D. Sontag, and C.L. Giles. The complexity of language recognition by neural networks. In J. van Leeuwen, editor, *Algorithms, Software, Architecture (Proceedings of IFIP* 12^{th} *World Computer Congress)*, pages 329–335, Amsterdam, 1992. North–Holland.

[34] M. Vidyasagar. Location and stability of the high–gain equilibria of nonlinear neural networks. *IEEE Transactions on Neural Networks*, 4(4):660–672, July 1993.

[35] X. Wang. Period-doublings to chaos in a simple neural network: An analytical proof. *Complex Systems*, (5):425–441, 1991.

[36] X. Wang and E.K. Blum. Discrete-time versus continuous-time models of neural networks. *Journal of Computer and Systems Sciences*, 45:1–19, 1990.

[37] R.L. Watrous and G.M. Kuhn. Induction of finite–state automata using second–order recurrent networks. In J.E. Moody, S.J. Hanson, and R.P. Lippmann, editors, *Advances in Neural Information Processing Systems 4*, pages 309–316, 1992.

[38] R.L. Watrous and G.M. Kuhn. Induction of finite–state languages using second–order recurrent networks. *Neural Computation*, 4(3):406–414, 1992.

[39] Z. Zeng, R.M. Goodman, and P. Smyth. Learning finite state machines with self–clustering recurrent networks. *Neural Computation*, 5(6):976–990, 1993.