Understanding State Space Organization in Recurrent Neural Networks with Iterative Function Systems Dynamics

Peter Tiňo[1,2], Georg Dorffner[1,3], and Christian Schittenkopf[1]

[1] Austrian Research Institute for Artificial Intelligence,
Schottengasse 3, A-1010 Vienna, Austria
{petert, georg, chris}@ai.univie.ac.at

[2] Department of Computer Science and Engineering, Slovak University of Technology,
Ilkovicova 3, 812 19 Bratislava, Slovakia

[3] Dept. of Medical Cybernetics and Artificial Intelligence, University of Vienna,
Freyung 6, A-1010 Vienna, Austria

In: *Hybrid Neural Symbolic Integration*, (eds) S. Wermter, R. Sun. pp. 256-270,
Springer Verlag, 2000.

Abstract. We study a novel recurrent network architecture with dynamics of iterative function systems used in chaos game representations of DNA sequences [16, 11]. We show that such networks code the temporal and statistical structure of input sequences in a strict mathematical sense: generalized dimensions of network states are in direct correspondence with statistical properties of input sequences expressed via generalized Rényi entropy spectra. We also argue and experimentally illustrate that the commonly used heuristic of finite state machine extraction by network state space quantization corresponds in this case to variable memory length Markov model construction.

1 Introduction

The correspondence between iterative function systems (IFS) [1] and recurrent neural networks (RNNs) has been recognized for some time [13, 28]. Because of the non-linear nature of RNN dynamics a deeper insight into RNN state space structure has been lacking. Also, even though there is a strong empirical evidence supporting usefulness of extracting finite state machines from recurrent networks trained on symbolic sequences [23, 7], we do not have a deeper understanding of what the machines actually represent.

We address these issues in the context of a novel recurrent network architecture, that we call iterative function system network[1] (IFSN). Dynamics of IFSNs

[1] Recently, we discovered that Tabor [28] independently investigated similar types of recurrent networks. However, while we are mainly interested in learning and representational issues in recurrent networks with IFS dynamics, Tabor's view is more general, with an emphasis on metric relations between the network representations of various forms of automata.

corresponds to iterative function systems used in chaos game representations of symbolic sequences [16, 11]. Using tools from multifractal theory and statistical mechanics we establish a rigorous relationship between statistical properties of sequences driving the network input and scaling behavior of IFSN states. We also analyse the structure of the IFSN state space and interpret the state space quantization leading to machine extraction in a Markovian context. We finish the paper by a detailed study of finite state machine extraction from IFSNs driven by the chaotic Feigenbaum sequence.

2 Formal Definitions

Consider a finite alphabet $\mathcal{A} = \{1, 2, ..., A\}$. The sets of all finite[2] and infinite sequences over $\mathcal{A}$ are denoted by $\mathcal{A}^+$ and $\mathcal{A}^\omega$ respectively. The set of all sequences consisting of a finite, or an infinite number of symbols from $\mathcal{A}$ is then $\mathcal{A}^\infty = \mathcal{A}^+ \cup \mathcal{A}^\omega$. The set of all sequences over $\mathcal{A}$ with exactly n symbols (i.e. of length n) is denoted by $\mathcal{A}^n$.

Let $S = s_1 s_2 ... \in \mathcal{A}^\infty$ and $i \leq j$. By S_i^j we denote the string $s_i s_{i+1} ... s_j$, with $S_i^i = s_i$.

2.1 Geometric Representations of Symbolic Sequence Structure

In this section we describe iterative function systems (IFSs) [1] acting on the N-dimensional unit hypercube $X = [0, 1]^N$, where[3] $N = \lceil \log_2 A \rceil$. To keep the notation simple, we slightly abuse mathematical notation and, depending on the context, regard the symbols $1, 2, ..., A$, as integers, or as referring to maps on X. The maps $i = 1, 2, ..., A$, constituting the IFS are affine contractions

$$i(x) = kx + (1 - k)t_i, \quad t_i \in \{0, 1\}^N, \quad t_i \neq t_j \text{ for } i \neq j, \tag{1}$$

with contraction coefficients $k \in (0, \frac{1}{2}]$.

The attractor of the IFS (1) is the unique set $K \subseteq X$, known as the Sierpinski sponge [12], for which $K = \bigcup_{i=1}^{A} i(K)$ [1].

For a string $u = u_1 u_2 ... u_n \in \mathcal{A}^n$ and a point $x \in X$, the point

$$u(x) = u_n(u_{n-1}(...(u_2(u_1(x)))...)) = (u_n \circ u_{n-1} \circ ... \circ u_2 \circ u_1)(x) \tag{2}$$

is considered a geometrical representation of the string u under the IFS (1). For a set $Y \subseteq X$, $u(Y)$ is then $\{u(x) |\ x \in Y\}$.

Denote the center $\{\frac{1}{2}\}^N$ of the hypercube X by x_*. Given a sequence $S = s_1 s_2 ... \in \mathcal{A}^\infty$, its (generalized) *chaos game representation* is formally defined as the sequence of points[4]

$$CGR_k(S) = \left\{ S_1^i(x_*) \right\}_{i \geq 1}. \tag{3}$$

[2] excluding the empty word

[3] for $x \in \Re$, $\lceil x \rceil$ is the smallest integer y, such that $y \geq x$

[4] the subscript k in $CGR_k(S)$ identifies the contraction coefficient of the IFS used for the geometric sequence representation

When $k = \frac{1}{2}$ and $\mathcal{A} = \{1,2,3,4\}$, we recover the IFS used by Jeffrey and others [11, 22, 25] to construct the chaos game representation of DNA sequences.

2.2 Statistics on Symbolic Sequences

Let $S = s_1 s_2 ... \in \mathcal{A}^\infty$ be a sequence generated by a stationary information source. Denote the (empirical) probability of finding an n-block $w \in \mathcal{A}^n$ in S by $P_n(w)$. A string $w \in \mathcal{A}^n$ is said to be an allowed n-block in the sequence S, if $P_n(w) > 0$. The set of all allowed n-blocks in S is denoted by $[S]_n$.

A measure of n-block uncertainty (per symbol) in S is given by the entropy rate

$$h_n(S) = -\frac{1}{n} \sum_{w \in [S]_n} P_n(w) \log P_n(w). \tag{4}$$

If information is measured in bits, then $\log \equiv \log_2$. The limit entropy rate $h(S) = \lim_{n\to\infty} h_n(S)$ quantifies the predictability of an added symbol (independent of block length).

The entropy rates h_n are special cases of Rényi entropy rates [24]. The β-order ($\beta \in \Re$) Rényi entropy rate

$$h_{\beta,n}(S) = \frac{1}{n(1-\beta)} \log \sum_{w \in [S]_n} P_n^\beta(w) \tag{5}$$

computed from the n-block distribution reduces to the entropy rate $h_n(S)$ when $\beta = 1$ [9]. The formal parameter β can be thought of as the inverse temperature in the statistical mechanics of spin systems [6]. In the infinite temperature regime, $\beta = 0$, the Rényi entropy rate $h_{0,n}(S)$ is just a logarithm of the number of allowed n-blocks, divided by n. The limit $h_{(0)}(S) = \lim_{n\to\infty} h_{0,n}(S)$ gives the asymptotic exponential growth rate of the number of allowed n-blocks, as the block length increases.

The entropy rates $h(S) = h_{(1)}(S) = \lim_{n\to\infty} h_{1,n}(S)$ and $h_{(0)}(S)$ are also known as the metric and topological entropies respectively.

Varying the parameter β amounts to scanning the original n-block distribution P_n – the most probable and the least probable n-blocks become dominant in the positive zero ($\beta = \infty$) and the negative zero ($\beta = -\infty$) temperature regimes respectively. Varying β from 0 to ∞ amounts to a shift from all allowed n-blocks to the most probable ones by accentuating still more and more probable subsequences. Varying β from 0 to $-\infty$ accentuates less and less probable n-blocks with the extreme of the least probable ones.

2.3 Scaling Behavior on Multifractals

Loosely speaking, a multifractal is a fractal set supporting a probability measure [2]. The degree of fragmentation of the fractal support M is usually quantified through its fractal dimension $D(M)$ [1]. Denote by $N(\ell)$ the minimal number of hyperboxes of side length ℓ needed to cover M. The fractal (box-counting) dimension $D(M)$ relates the side length ℓ with $N(\ell)$ via the scaling law $N(\ell) \approx \ell^{-D(M)}$.

For $0 < k \leq \frac{1}{2}$, the n-th order approximation $D_{n,k}(M)$ of the fractal dimension $D(M)$ is given by the box-counting technique with boxes of side $\ell = k^n$:

$$N(k^n) = (k^n)^{-D_{n,k}(M)} .$$

Just as the Rényi entropy spectra describe (non-homogeneous) statistics on symbolic sequences, generalized Rényi dimensions D_β capture multifractal probabilistic measures μ [15]. Generalized dimensions $D_\beta(M)$ of an object M describe a measure μ on M through the scaling law

$$\sum_{B \in \mathcal{B}_\ell,\ \mu(B)>0} \mu^\beta(B) \approx \ell^{(\beta-1)D_\beta(M)}, \tag{6}$$

where $\mathcal{B}_\ell$ is a minimal set of hyperboxes with sides of length ℓ disjointly[5] covering M.

In particular, for $0 < k \leq \frac{1}{2}$, the n-th order approximation $D_{\beta,n,k}(M)$ of $D_\beta(M)$ is given by

$$\sum_{B \in \mathcal{B}_\ell,\ \mu(B)>0} \mu^\beta(B) = \ell^{(\beta-1)D_{\beta,n,k}(M)}, \tag{7}$$

where $\ell = k^n$.

The infinite temperature scaling exponent $D_0(M)$ is equal to the box-counting fractal dimension $D(M)$ of M. Dimensions D_1 and D_2 are respectively known as the information and correlation dimensions [2]. Of special importance are the limit dimensions D_∞ and $D_{-\infty}$ describing the scaling behavior of regions where the probability is most concentrated and rarefied respectively.

2.4 Chaos Game Representation of Single Sequences

In [16] we established a relationship between the Rényi entropy spectra of a sequence $S \in \mathcal{A}^\infty$ and the generalized dimension spectra of its chaos game representations.

Theorem 1 [16]: *For any sequence $S \in \mathcal{A}^\infty$, and any $n = 1, 2, ...,$ the n-th order approximations of the generalized dimensions of its game representations are equal (up to a scaling constant $\log k^{-1}$) to the sequence n-block Rényi entropy rate estimates:*

$$D_{\beta,n,k}(CGR_{n,k}(S)) = \frac{h_{\beta,n}(S)}{\log \frac{1}{k}}, \tag{8}$$

where $CGR_{n,k}(S)$ is the sequence $CGR_k(S)$ without the first $n-1$ points. Furthermore, for each $S \in \mathcal{A}^\omega$,

$$D_{\beta,n,k}(CGR_k(S)) = \frac{h_{\beta,n}(S)}{\log \frac{1}{k}}. \tag{9}$$

[5] at most up to Lebesgue measure zero borders

Hence, for infinite sequences $S \in \mathcal{A}^\omega$, when $k = \frac{1}{2}$, the generalized dimension estimates of geometric chaos game representations exactly equal the corresponding sequence Rényi entropy rate estimates. In particular, given an infinite sequence $S \in \mathcal{A}^\omega$, as n grows, the box-counting fractal dimension and the information dimension estimates $D_{0,n,\frac{1}{2}}$ and $D_{1,n,\frac{1}{2}}$ of the original Jeffrey chaos game representation [11, 22, 25] tend to the sequence topological and metric entropies respectively.

Another nice property of the chaos game representation CGR_k is that it codes the suffix structure of allowed subsequences in the distribution of subsequence geometric representations (2) [16]. In particular, if $v \in \mathcal{A}^+$ is a suffix of length $|v|$ of a string $u = rv$, $r, u \in \mathcal{A}^+$, then $u(X) \subset v(X)$, where $v(X)$ is an N-dimensional hypercube of side length $k^{|v|}$. Hence, the longer is the common suffix v shared by two subsequences rv and $rvqv$ of a sequence $S = rvqvw$, $r, q, v \in \mathcal{A}^+$, $w \in \mathcal{A}^\infty$, the closer lie the corresponding points $rv(x_*)$ and $rvqv(x_*)$ in the chaos game representation of S,

$$d_E\left(rv(x_*), rvqv(x_*)\right) \leq k^{|v|}\sqrt{N}. \tag{10}$$

Here d_E denotes the Euclidean distance.

3 Recurrent Neural Network

Recurrent neural network (RNN) with adjustable recurrent weights presented in figure 1 was shown to be able to learn mappings that can be described by finite state machines [20], or produce symbolic sequences closely approximating (with respect to the information theoretic entropy and cross-entropy measures) the statistical structure in long chaotic training sequences [21, 19].

The network has an input layer $I^{(t)} = (I_1^{(t)}, ..., I_A^{(t)})$ with A neurons (to which the one-of-A codes of input symbols from the alphabet $\mathcal{A} = \{1, ..., A\}$ are presented, one at a time), a hidden non-recurrent layer $H^{(t)}$, a hidden recurrent layer $R^{(t+1)}$ (RNN state space), and an output layer $O^{(t)}$ having the same number of neurons as the input layer. Activations in the recurrent layer are copied with a unit time delay to the context layer $R^{(t)}$ that forms an additional input.

Using second-order hidden units, at each time step t, the input $I^{(t)}$ and the context $R^{(t)}$ determine the output $O^{(t)}$ and the future context $R^{(t+1)}$ by

$$H_i^{(t)} = g\left(\sum_{j,k} Q_{i,j,k} I_j^{(t)} R_k^{(t)} + T_{i,j}^H\right), \tag{11}$$

$$O_i^{(t)} = g\left(\sum_{j} V_{i,j} H_j^{(t)} + T_i^O\right), \tag{12}$$

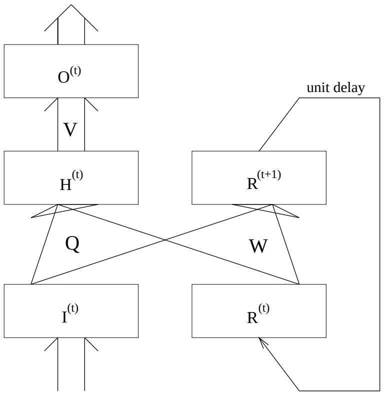

Fig. 1. Recurrent neural network (RNN) architecture. When recurrent weights W and thresholds T^R are fixed prior to the training process and activation functions in the recurrent layer $R^{(t+1)}$ are linear so that the recurrent part $[I^{(t)} + R^{(t)} \rightarrow R^{(t+1)}]$ of the network implements the IFS (1), the architecture is referred as iterative function system network (IFSN).

$$R_i^{(t+1)} = g\left(\sum_{j,k} W_{i,j,k} I_j^{(t)} R_k^{(t)} + T_{i,j}^R\right). \tag{13}$$

Here, g is the standard logistic sigmoidal function. $W_{i,j,k}$, $Q_{i,j,k}$ and $T_{i,j}^R$, $T_{i,j}^H$ are second-order real valued weights and thresholds, respectively. $V_{i,j}$ and T_i^O are the weights and thresholds, respectively, associated with the hidden to output layer connections.

When first-order hidden units are used, eqs. (11) and (13) change to

$$H_i^{(t)} = g\left(\sum_j Q_{i,j}^I I_j^{(t)} + \sum_k Q_{i,k}^R R_k^{(t)} + T_i^H\right) \tag{14}$$

and

$$R_i^{(t+1)} = g\left(\sum_j W_{i,j}^I I_j^{(t)} + \sum_k W_{i,k}^R R_k^{(t)} + T_i^R\right), \tag{15}$$

respectively.

3.1 Previous Work

We briefly describe our previous experiments [21, 19] with the RNN introduced above. The network was trained (via Real Time Recurrent Learning [10]) on single long chaotic symbolic sequences $S = s_1 s_2 ...$ to predict, at each point in time, the next symbol. To start the training, the initial network state $R^{(1)}$ was randomly generated and the network was reset with $R^{(1)}$ at the beginning of each training epoch. After the training, the network was seeded with the initial state $R^{(1)}$ and code of the first symbol s_1 in S. Then the network acted as a stochastic source. We transformed the RNN output $O^{(t)} = (O_1^{(t)}, ..., O_A^{(t)})$ into a probability distribution over symbols $\hat{s}_{t+1}$ that will appear at the net input at the next time step:

$$Prob\,(\hat{s}_{t+1} = i) = \frac{O_i^{(t)}}{\sum_{j=1}^{A} O_j^{(t)}}, \quad i = 1, 2, ..., A. \tag{16}$$

We observed that trained RNNs produced sequences closely mimicking (in the information theoretic sense) the training sequence.

Next we extracted from trained RNNs stochastic finite state machines by identifying clusters in recurrent neurons' activation space (RNN state space) with states of the extracted machines. The extracted machines provide a compact and easy-to-analyse symbolic representation of the knowledge induced in RNNs during the training. Finite machine extraction from RNNs is a commonly used heuristic especially among recurrent network researchers applying their models in grammatical inference tasks [23]. The extracted machines often outperform the original networks on longer test strings. For an analysis of this phenomenon see [4, 18]. In our experiments we found that these issues translate to the problem of training RNNs on single long chaotic sequences. With sufficient number of states the extracted stochastic machines do indeed replicate the entropy and cross-entropy performance of their mother RNNs.

We considered two principal ways of machine extraction. In the *test mode extraction* we let the RNN generate a sequence in an autonomous mode and code the transitions between quantized network states driven by the generated symbols as stochastic[6] machines M_{RNN}. In the *training sequence driven construction* we drive the RNN with the training sequence S and code the transitions between quantized RNN states on symbols in S as stochastic machines $M_{RNN(S)}$. We found that the machines $M_{RNN(S)}$ achieved considerably better modeling performance than their mother RNNs [19].

3.2 Recurrent Neural Network with IFS Dynamics

We propose an alternative RNN architecture, which we call *iterative function system network* (IFSN). IFSNs share the architecture with RNNs described above (see figure 1), with the exception that the recurrent neurons' activation function is linear and the weights W (W^I, W^R) and thresholds T^R are *fixed*, so that the network dynamics in eq. (13) (eq. (15)) is given by (1): given the current state $R^{(t)}$, $i(R^{(t)})$ is the next state $R^{(t+1)}$, provided the input symbol at time t is i. Such a dynamics is equivalent to the dynamics of the IFS (1) driven by the

[6] with each state transition in the machine we associate its empirical probability by counting how often was that transition evoked during the extraction process

symbolic sequence appearing at the network input. The trainable parts of IFSNs form a feed-forward architecture $[I^{(t)} + R^{(t)} \rightarrow H^{(t)} \rightarrow O^{(t)}]$.

In our recent experimental study [17] on two long chaotic sequences with different degrees of "complexity" measured by Crutchfield's ϵ-machines [5], we have (interestingly enough) found that even though IFSNs have fixed non-trainable recurrent parts, they achieved performances comparable with those of RNNs having adjustable recurrent weights and thresholds. Moreover, the extracted machines $M_{IFSN(S)}$ actually outperformed the corresponding machines $M_{RNN(S)}$.

4 Understanding State Space Organization in IFSNs

Although previous approaches to the analysis of RNN state space organization did point out the correspondence between IFSs and RNN recurrent part $[I^{(t)} + R^{(t)} \rightarrow R^{(t+1)}]$ [14, 13], due to nonlinearity of recurrent neurons' activation function, they did not manage to provide a deeper insight into the RNN state space structure (apart from observing an apparent fractal-like clusters corresponding to nonlinear IFS attractor). Also, despite strong empirical evidence supporting the usefulness of extracting symbolic finite state representations from trained recurrent networks [23, 7] a deeper understanding of what the machines actually represent is still lacking.

The results summarized in section 2.4 enable us to formulate the principles behind coding, in the recurrent part of IFSNs, of the temporal/statistical structure in symbolic sequences appearing at the network input. Theorem 1 tells us that the fractal dimension of states in IFSN driven by a sequence S directly corresponds to the allowed subsequence variability in S expressed through the topological entropy of S. An analogical relationship holds between the information dimension of IFSN states and the metric entropy of S. Generally, multifractal characteristics of IFSN states measured via spectra of generalized dimensions directly correspond to Rényi entropy spectra of the input sequence S.

The input sequence S feeding the IFSN is translated in the network recurrent part into the chaos game representation $CGR_k(S)$. The $CGR_k(S)$ forms clusters of state neuron activations, where (as explained in section 2.4) points lying in a close neighborhood code histories with a long common suffix (e.g. histories that are likely to produce similar continuations), whereas histories with different suffices (and potentially different continuations) are mapped to activations lying far from each other. When quantizing the IFSN state space to extract the network finite state representation, densely populated areas (corresponding to contexts with long common suffices) are given more attention by the vector quantizer. Consequently, more information processing states of the extracted machines are devoted to these potentially "problematic" contexts. This directly corresponds to the idea of variable memory length Markov models [26, 27], where the length of the past history considered in order to predict the future is not fixed, but context dependent.

5 Extracting Finite State Stochastic Machines from IFSNs

We report an experiment with a well-known chaotic sequence, called the Feigenbaum sequence [8]. Feigenbaum sequence is a binary sequence generated by the logistic map $y_{t+1} = ry_t(1-y_t)$, $y \in [0,1]$, with the control parameter r set to the period doubling accumulation point value[7] [15]. The iterands y_t are partitioned into two regions[8] $[0, \frac{1}{2})$ and $[\frac{1}{2}, 1]$, corresponding to symbols 1 and 2, respectively. The training sequence S used in this study contained 260.000 symbols.

The topological structure of the sequence (i.e. the structure of allowed n-blocks not regarding their probabilities) can only be described using a context sensitive tool – a restricted indexed context-free grammar [6]. The metric structure of the Feigenbaum sequence is organized in a self-similar fashion [8]. The transition between the ranked distributions for block lengths $2^g \to 2^{g+1}$, $3\ 2^{g-1} \to 3\ 2^g$, $g \geq 1$, is achieved by rescaling the horizontal and vertical axis by a factor 2 and $\frac{1}{2}$, respectively. Plots of the Feigenbaum sequence n-block distributions, $n = 1, 2, ..., 8$, can be seen in figure 2. Numbers above the plots indicate the corresponding block lengths. The arrows connect distributions with the $(2, \frac{1}{2})$-scaling self-similarity relationship.

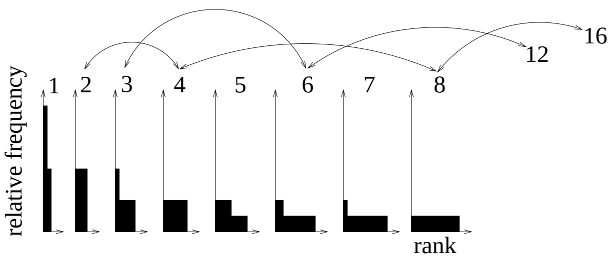

Fig. 2. Plots of self-similar rank-ordered block distributions of the Feigenbaum sequence for different block lengths (indicated by the numbers above the plots). The self similarity relates block distributions for block lengths $2^g \to 2^{g+1}$, $3\ 2^{g-1} \to 3\ 2^g$, $g \geq 1$ (connected by arrows).

We chose to work with the Feigenbaum sequence because Markovian predictive models on this sequence need deep prediction contexts. Classical fixed-order Markov models (MMs) cannot succeed and the power of admitting a limited number of variable length contexts can be fully exploited.

First, we built a series of variable memory length Markov models (VLMMs) of growing size. For construction details see [26, 27]. Then, we quantized the

[7] r=3.56994567...

[8] this partition is a generating partition defined by the critical point

one-dimensional[9] IFSN state space[10] using dynamic cell structures (DCS) technique [3]. This way we obtained a series of extracted machines $M_{IFSN(S)}$ with increasing number of machine states.

Each model was used to generate 10 sequences G of length equal to the length of the training sequence. Since the Feigenbaum sequence n-block distributions have just one or two probability levels, we measure the disproportions between the Feigenbaum and model generated distributions through the L_1 distances, $d_n(S,G) = \sum_{w \in \mathcal{A}^n} |P_{S,n}(w) - P_{G,n}(w)|$, where $P_{S,n}$ and $P_{G,n}$ are the empirical n-block frequencies in the training and model generated sequences, respectively.

A modeling horizon $n(\mathcal{M})$ of a model $\mathcal{M}$ is the longest block length, such that, for all 10 sequence generation realizations and for all block lengths $n \leq n(\mathcal{M})$, $d_n(S,G)$ is below a small threshold Δ. We set $\Delta = 0.005$, since in this experiment, either $d_n(S,G) \in (0, 0.005]$, or $\delta_n(\mathcal{M}) >> 0.005$.

Figure 3 interprets the growing ability of VLMMs and machines $M_{IFSN(S)}$ to model the metric structure of allowed blocks in the Feigenbaum sequence S.

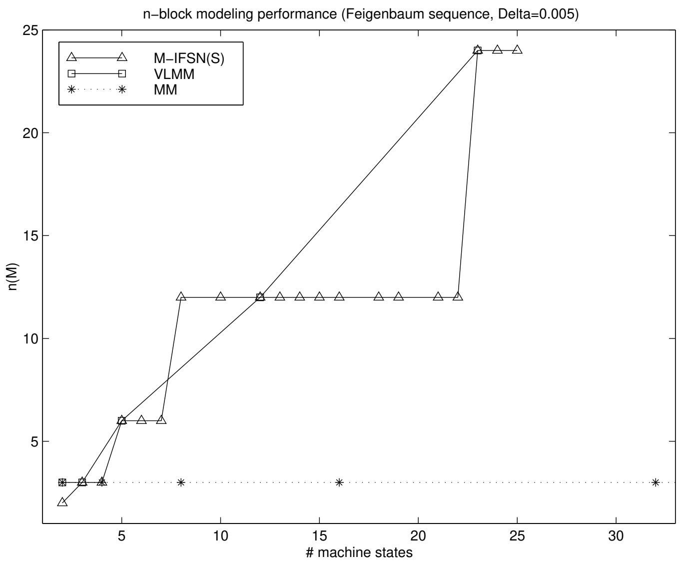

Fig. 3. Modeling horizons $n(\mathcal{M})$ of models $\mathcal{M}$ built on the Feigenbaum sequence as a function of the number machine states in $\mathcal{M}$.

The classical MM (stars) totally fails in this experiment, since the context length 5 is far too small to enable the MM to mimic the complicated subsequence

[9] Chaos representation of binary sequences is defined by a one-dimensional IFS (1). In this experiment we set the contraction ratio k of maps in the IFS to 0.5.

[10] Note that since the training sequence driven extraction of machines $M_{IFSN(S)}$ uses only recurrent part of IFSN (which is fixed prior to the training), no network training is needed and the finite state representations $M_{IFSN(S)}$ can be readily constructed.

structure in S. On the other hand VLMMs (squares) and machines $M_{IFSN(S)}$ (triangles) quickly learn to explore a limited number of deep prediction contexts and perform comparatively well.

The jumps in the modeling horizon graph of machines $M_{IFSN(S)}$ on figure 3 can be understood through their state transition diagrams.

While the machine M_4 in figure 4a can model only blocks of length 1,2 and 3, the introduction of an additional transition state in the machine M_5 shown in figure 4b enables the latter machine to model blocks of length up to 6.

Only three consecutive 2's are allowed in the Feigenbaum training sequence S. The loop on symbol 2 in the state 1 of the machine M_4 is capable of producing blocks of consecutive 2's of any length. So, the n-block distribution, $n \geq 4$, cannot be properly modeled by the machine M_4. The state 1 in the machine M_4 is split into two machine M_5 states 1.a and 1.b. Any number of 4-blocks `2212` can be followed by any number of 2-blocks `12` and vice versa. This is fine as long as we study structure of the 6-block distribution.

Moving to higher block lengths, we find that once the 4-block `2212` is followed by the 2-block `12`, another copy of the 2-block `12` followed by the 4-block `2212` must appear. This 12-block rule is implemented by the machine M_8 in figure 5b. The machine M_8 is created from the machine M_7 in figure 5a by splitting the state 3.a into two states 3.a and 3.c. The machine M_7 with 7 states is equivalent to the machine M_5 (figure 4a) with 5 states: states 2.a, 2.b and 3.a, 3.b in M_7 are equivalent to states 2 and 3, respectively, in M_5.

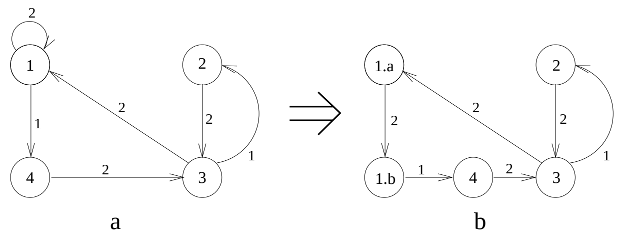

Fig. 4. State transition diagrams of the machines $M_{IFSN(S)}$. The machines M_4 (a) and M_5 (b) were obtained by quantizing IFSN state space via dynamic cell structures with 4 and 5 centers respectively. State transitions are labeled only with the corresponding symbols, since the transition probabilities are uniformly distributed, i.e. for all states i, the probability associated with each arc leaving i is equal to $1/N_i$, where N_i is the number of arcs leaving state i.

State splitting responsible for the third jump in the modeling horizon graph between the extracted machines M_{22} and M_{23} with 22 and 23 states, respectively, is illustrated in figure 6. Symbols `A` and `B` stand for the 4-blocks `1212` and `2212`, respectively. The machine M_{22} is equivalent to the machine M_8. State splitting in the middle left branch of the machine M_{22} removes the two lower cycles `BAB, B`, and creates a single larger cycle `BBAB` in the machine M_{23}. This machine correctly

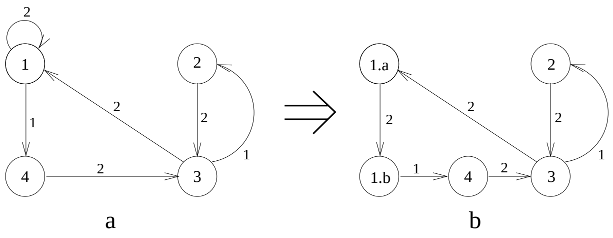

Fig. 5. Machines M_7 and M_8 extracted from IFSN driven by the Feigenbaum sequence S. The network state space is quantized into 7 (a) and 8 (b) compartments, respectively. Construction details are described in caption to the previous figure.

implements the training sequence block distributions for blocks of length up to 24.

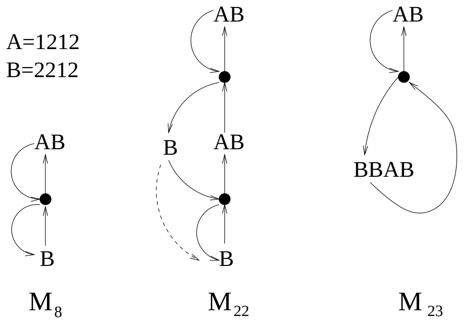

Fig. 6. Schematic representation of state transition structure in machines $M_{IFSN(S)}$. Symbols `A` and `B` stand for the 4-blocks `1212` and `2212`, respectively. The machine M_{22}, obtained from a codebook with 22 centers, is equivalent to the machine M_8 (see also the previous figure). State splitting in the middle left branch of the machine M_{22} (dashed line) removes the two lower cycles `BAB, B`, and creates a single larger cycle `BBAB` in the machine M_{23}.

Variable memory length Markov models implement the same subsequence constraints as the machines $M_{IFSN(S)}$. Figures 7a and 7b present VLMMs N_5 and N_{11} with 5 and 11 prediction contexts, respectively. The VLMMs are shown as probabilistic suffix automata with states labeled by the corresponding suffices. The VLMM N_5 is isomorphic to the machine M_5 in figure 4b, and the VLMM N_{11} is equivalent to the machine M_8 in figure 5b. Although not shown here, the VLMM with 23 prediction contexts is isomorphic to the machine M_{23} schematically presented in figure 6.

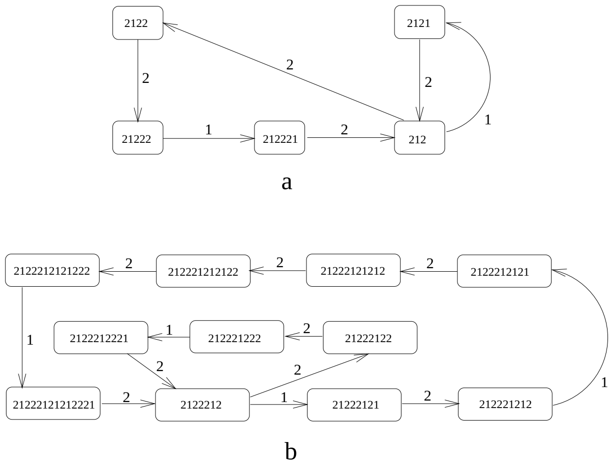

Fig. 7. VLMMs N_5 (a) and N_{11} (b) built on the Feigenbaum sequence. The VLMMs are are shown as probabilistic suffix automata with states labeled by the corresponding variable length prediction contexts. As with machines $M_{IFSN(S)}$ in this experiment, the state transition probabilities are uniformly distributed.

6 Conclusion

We introduced a novel recurrent network architecture, that we call iterative function system network (IFSN), with dynamics corresponding to iterative function systems used in chaos game representations of symbolic sequences [16, 11].

In our previous work on modeling long chaotic sequences we empirically compared recurrent networks having adjustable recurrent weights and non-linear sigmoid activations in the recurrent layer with IFSNs and showed that introducing fixed IFS dynamics into recurrent networks does not degradate the network performance. Even more surprisingly, we found that finite state stochastic machines extracted from IFSNs outperform machines extracted from "fully adjustable" RNNs.

In this contribution we formally study state space organization in IFSNs. It appears that IFSNs reflect in their states the temporal and statistical structure of input sequences in a strict mathematical sense. The generalized dimensions of IFSN states are in direct correspondence with statistical properties of input sequences expressed via generalized Rényi entropy spectra.

We also argued and experimentally illustrated that the commonly used heuristic of finite state machine extraction from RNNs by network state space quantization corresponds in case of IFSNs to variable memory length Markov model construction.

Acknowledgements

This work was supported by the Austrian Science Fund (FWF) within the research project "Adaptive Information Systems and Modeling in Economics and Management Science" (SFB 010). The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry of Science and Transport.

References

1. M.F. Barnsley. *Fractals everywhere.* Academic Press, New York, 1988.
2. C. Beck and F. Schlogl. *Thermodynamics of chaotic systems.* Cambridge University Press, Cambridge, UK, 1995.
3. J. Bruske and G. Sommer. Dynamic cell structure learns perfectly topology preserving map. *Neural Computation*, 7(4):845–865, 1995.
4. M.P. Casey. The dynamics of discrete-time computation, with application to recurrent neural networks and finite state machine extraction. *Neural Computation*, 8(6):1135–1178, 1996.
5. J.P. Crutchfield and K. Young. Inferring statistical complexity. *Physical Review Letters*, 63:105–108, July 1989.
6. J.P. Crutchfield and K. Young. Computation at the onset of chaos. In W.H. Zurek, editor, *Complexity, Entropy, and the physics of Information, SFI Studies in the Sciences of Complexity, vol 8*, pages 223–269. Addison-Wesley, Reading, Massachusetts, 1990.
7. P. Frasconi, M. Gori, M. Maggini, and G. Soda. Insertion of finite state automata in recurrent radial basis function networks. *Machine Learning*, 23:5–32, 1996.
8. J. Freund, W. Ebeling, and K. Rateitschak. Self-similar sequences and universal scaling of dynamical entropies. *Physical Review E*, 54(5):5561–5566, 1996.
9. P. Grassberger. Information and complexity measures in dynamical systems. In H. Atmanspacher and H. Scheingraber, editors, *Information Dynamics*, pages 15–33. Plenum Press, New York, 1991.
10. J. Hertz, A. Krogh, and R.G. Palmer. *Introduction to the Theory of Neural Computation.* Addison–Wesley, Redwood City, CA, 1991.
11. J. Jeffrey. Chaos game representation of gene structure. *Nucleic Acids Research*, 18(8):2163–2170, 1990.
12. R. Kenyon and Y. Peres. Measures of full dimension on affine invariant sets. *Ergodic Theory and Dynamical Systems*, 16:307–323, 1996.
13. J.F. Kolen. Recurrent networks: state machines or iterated function systems? In M.C. Mozer, P. Smolensky, D.S. Touretzky, J.L. Elman, and A.S. Weigend, editors, *Proceedings of the 1993 Connectionist Models Summer School*, pages 203–210. Erlbaum Associates, Hillsdale, NJ, 1994.
14. P. Manolios and R. Fanelli. First order recurrent neural networks and deterministic finite state automata. *Neural Computation*, 6(6):1155–1173, 1994.
15. J.L. McCauley. *Chaos, Dynamics and Fractals: an algorithmic approach to deterministic chaos.* Cambridge University Press, 1994.
16. P. Tiňo. Spatial representation of symbolic sequences through iterative function system. *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, 29(4):386–393, 1999.

17. P. Tiňo and G. Dorffner. Recurrent neural networks with iterated function systems dynamics. In *International ICSC/IFAC Symposium on Neural Computation*, pages 526–532, 1998.
18. P. Tiňo, B.G. Horne, C.L. Giles, and P.C. Collingwood. Finite state machines and recurrent neural networks – automata and dynamical systems approaches. In J.E. Dayhoff and O. Omidvar, editors, *Neural Networks and Pattern Recognition*, pages 171–220. Academic Press, 1998.
19. P. Tiňo and M. Koteles. Extracting finite state representations from recurrent neural networks trained on chaotic symbolic sequences. *IEEE Transactions on Neural Networks*, 10(2):284–302, 1999.
20. P. Tiňo and J. Sajda. Learning and extracting initial mealy machines with a modular neural network model. *Neural Computation*, 7(4):822–844, 1995.
21. P. Tiňo and V. Vojtek. Modeling complex sequences with recurrent neural networks. In G.D. Smith, N.C. Steele, and R.F. Albrecht, editors, *Artificial Neural Networks and Genetic Algorithms*, pages 459–463. Springer Verlag Wien New York, 1998.
22. J.L. Oliver, P. Bernaola-Galván, J. Guerrero-Garcia, and R. Román Roldan. Entropic profiles of dna sequences through chaos-game-derived images. *Journal of Theor. Biology*, (160):457–470, 1993.
23. C.W. Omlin and C.L. Giles. Extraction of rules from discrete-time recurrent neural networks. *Neural Networks*, 9(1):41–51, 1996.
24. A. Renyi. On the dimension and entropy of probability distributions. *Acta Math. Hung.*, (10):193, 1959.
25. R. Roman-Roldan, P. Bernaola-Galvan, and J.L. Oliver. Entropic feature for sequence pattern through iteration function systems. *Pattern Recognition Letters*, 15:567–573, 1994.
26. D. Ron, Y. Singer, and N. Tishby. The power of amnesia. In *Advances in Neural Information Processing Systems 6*, pages 176–183. Morgan Kaufmann, 1994.
27. D. Ron, Y. Singer, and N. Tishby. The power of amnesia. *Machine Learning*, 25:117–150, 1996.
28. W. Tabor. Dynamical automata. Technical Report TR98-1694, Cornell University, Computer Science Department, 1998.