ESANN'2007 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2007, d-side publi., ISBN 2-930307-07-2.

Structured reservoir computing with spatiotemporal chaotic attractors

Carlos Lourenço[1,2] *

1- Faculty of Sciences of the University of Lisbon - Informatics Department
Campo Grande, 1749-016 Lisboa - Portugal

2- Instituto de Telecomunicações - Security and Quantum Information Group
Av. Rovisco Pais, 1, 1049-001 Lisboa - Portugal

Abstract. We approach the themes "computing with chaos" and "reservoir computing" in a unified setting. Different neural architectures are mentioned which display chaotic as well as reservoir properties. The architectures share a common topology of close-neighbor connections which supports different types of spatiotemporal dynamics in continuous time. We bring up the role of spatiotemporal structure and associated symmetries in reservoir-mediated pattern processing. Such type of computing is somewhat different from most other examples of reservoir computing.

1 Spatiotemporal chaotic reservoirs

Previously we have described the reservoir property in the context of chaotic dynamical systems and pointed out that it is a useful manifestation of chaos' flexibility in view of computation [1, 2, 3, 4]. The recent developments in reservoir computing research [5, 6, 7] prompted us to recall some of the previous ideas concerning computing with dynamical networks, as well as to provide new results and suggest research directions. Our work aims at classifying arbitrarily complex time-varying patterns, where time is continuous. We depart from the mainstream of reservoir computing systems in that our neural network is topologically organized through close-neighbor connections. In the literature the latter arrangement is sometimes called a cellular neural network. The connection topology makes the network particularly suitable for processing spatiotemporal input patterns. We emphasize that the resulting input-output relation (or "filter") can in certain cases be understood in terms of spatiotemporal symmetries inherent to the chaotic attractor and the Unstable Periodic Orbits (UPOs) therein. Such symmetries can be tuned to the particular needs of input classification tasks.

Our earlier description of the spatiotemporal regimes involved in this so-called structured reservoir computing shows that computation happens at the edge of chaos, specifically on the chaotic side [2, 4]. Chaos is not fully developed, and indeed it appears to display just the right balance between dynamical structure and flexibility. Hence, the adjective "structured" when applied to the

*The author acknowledges the partial support of Fundação para a Ciência e a Tecnologia and EU FEDER via the Center for Logic and Computation and the project ConTComp (POCTI/MAT/45978/2002), and also via the project PDCT/MAT/57976/2004. Research performed under the scope of project RealNComp (PTDC/MAT/76287/2006).

reservoir can refer both to the structure of the chaotic attractor and to the structure of connections between the neurons supporting the attractor.

The emphasis on the adequacy for spatiotemporal input patterns might be confused for a dismissal of the general-purpose nature of reservoir computing. However, nothing prevents the usage of the proposed reservoir in the processing of time-varying patterns without noticeable spatial structure, including e.g. the extreme cases of a scalar time-series or an array of values constant in time.

We take advantage of the chaotic attractors' property of possessing an infinite number of embedded UPOs [2, 4], to be used as computational modes suitable for particular tasks. This usage of the attractor's structure may be associated with a control mechanism able to stabilize particular UPOs very fast and with minimum perturbation of the original system. The stabilization places the system in a state appropriate for the processing of a certain type of input patterns, and need only last the minimum amount of time required for computation to occur. Such control may also be absent, but nevertheless the attractor's structure may be revealed in the way the system responds to particular input patterns. In [4] the ensemble of UPOs was itself called a "reservoir", following previous nomenclature [1]. On the other hand, the range of dynamical input-output relations made possible by chaos' flexibility could also be viewed as a reservoir [4], in the sense of a functional reservoir.

An obvious common point of our approach with mainstream reservoir computing is that the dynamics of a pool of neurons in a recurrent network is perturbed by some input stimulus (the "pattern") and the resulting transient dynamics is assessed by a separate readout mechanism as a process which can provide meaning to the observed dynamics. The system is perturbed differently according to specific characteristics of each input, and features a fading memory. We allow the chaotic attractor to be perturbed either in a permanent or in a transient fashion, depending on the task's needs. So far we have not implemented a learning procedure for an explicit readout layer of neurons, but were nonetheless able to obtain sets of dynamical response functions adequate for pattern classification, that is, easily separable.

2 Leaky-integrator reservoir

2.1 Setting up the network

In the present paper we consider mainly the biologically inspired model described in detail in [3, 4]. Let us briefly recall the model, while at the same time extending it to support an instance of reservoir computing. After having investigated many different variations of network architectures with one and two spatial dimensions, featuring different network sizes and intrinsic parameter values leading to chaos, we focused on a 1-D spatial neural arrangement that allows the system to be situated in an appropriate edge of chaos regime in the spirit of the previous section. A population of N excitatory neurons X_i and N inhibitory neurons Y_i is coupled as shown in Fig. 1. The evolution of the membrane potentials X_i and

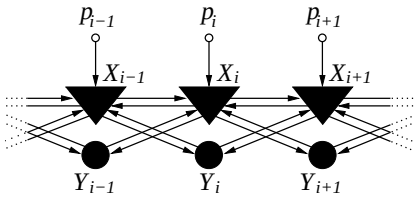

Fig. 1: Neural network with excitatory neurons X_i (triangles) and inhibitory neurons Y_i (large circles) connected via $X_i \to X_{i\pm1}$ and $X_i \to Y_{i\pm1}$ excitatory connections, and via $Y_i \to X_{i\pm1}$ inhibitory connections. Shown also are the external inputs p_i, in a one-to-one correspondence with the excitatory neurons.

Y_i in the absence of any perturbation is described by the following equations:

$$\begin{aligned}
\frac{dX_i}{dt} &= -\gamma(X_i - V_L) - (X_i - E_1) \sum_{j=i\pm1} \omega_{ij}^{(1)} F_X[X_j(t-\tau_{ij})] \\
&\quad - (X_i - E_2) \sum_{j=i\pm1} \omega_{ij}^{(2)} F_Y[Y_j(t-\tau_{ij})] \\
\frac{dY_i}{dt} &= -\gamma(Y_i - V_L) - (Y_i - E_1) \sum_{j=i\pm1} \omega_{ij}^{(3)} F_X[X_j(t-\tau_{ij})] \\
&\quad i = 1, \ldots, N \, .
\end{aligned}$$

The resting potential V_L has the value -60 mV, whereas the ionic potentials have the respective values $E_1 = 50$ mV and $E_2 = -80$ mV. The inverse of the membrane's time-constant takes the value $\gamma = 0.25$ msec^{-1}. The neurons are nonlinearly coupled through sigmoidal functions $F(V) = 1/\left(1 + \exp\left[-\alpha(V - V_c)\right]\right)$ where V stands for X or Y. α takes different values depending on whether an excitatory or an inhibitory neuron is implied: $\alpha_X = 0.09$ mV^{-1} and $\alpha_Y = 0.2$ mV^{-1}. The activation threshold is dependent upon V_c which is fixed at -25 mV. We consider a homogeneous time-delay $\tau_{ij} = \tau = 1.8$ msec due to the finite speed of signal propagation between neurons. The synaptic weights ω are also homogeneously chosen within each type of connection, and they are constant in time. The excitatory-to-excitatory weights $\omega_{ij}^{(1)} = \omega^{(1)}$ have the value 3.15; the inhibitory-to-excitatory weights $\omega_{ij}^{(2)} = \omega^{(2)}$ have the value 1.64; the excitatory-to-inhibitory weights $\omega_{ij}^{(3)} = \omega^{(3)}$ have the value 2.5; no inhibitory-to-inhibitory connections are considered. In accordance with Fig. 1, the network has first-neighbor connectivity. The boundary conditions are of the zero-flux type. For the present numerical simulations we take $N = 8$.

An account of the unperturbed chaotic regime and UPO reservoir properties of this model was given in [4], where several UPOs were identified. The latter display diverse spatiotemporal symmetries. Here we extend the model by considering an N-dimensional array of external perturbations $p_i(t)$, where each p_i perturbs at most one neuron at the corresponding position, namely X_i. Each p_i

may vary continuously over time t. Depending on correlations that may exist between the p_i for different i, the collective evolution of the set of p_i during some time-interval may be viewed as a true spatiotemporal pattern. However, positive correlations between, say, adjacent p_i are not mandatory for reservoir computing to take place. In any occurrence, the underlying spatiotemporal symmetries of the attractor being perturbed are bound to play an important role in the generation of output signals.

As described in [4], the neurons display individual differences in their oscillatory activity, with a partial temporal and spatial correlation kept throughout the unperturbed chaotic regime. Monitoring the evolution of, say, the profile of simultaneous membrane potentials of excitatory neurons, allows us to gather all these neurons into a single spatiotemporal evolution. The latter is chaotic, but to some degree it displays traces of the UPOs embedded in the attractor. Each of these UPOs is strictly periodic and is itself a spatiotemporal object with a defined symmetry. Some of the UPOs may present e.g. a left-to-right symmetry with respect to spatial position, whereas others do not display such symmetry. In the absence of a control mechanism, none of these UPOs can be spontaneously sustained. However, by waiting long enough, an arbitrarily small vicinity of each of the UPOs may be attained.

2.2 Pattern processing

We propose three variations of reservoir computing with chaotic neural networks where the networks are generically of the above type, namely featuring a regular spatial arrangement and close-neighbor connectivity. Note that the networks in question need not be exactly as described in Section 2.1. For lack of space, we briefly refer the reader to an application of these ideas with an alternative neural network [2] for which the chosen examples of input patterns happen to be more complex (and dynamical) than the ones shown in the present paper.

1. In a network with two identical chaotic layers of neurons, the first layer is stabilized into a certain UPO which is task-dependent, for the duration of input processing. The second layer is perturbed by a spatiotemporal input pattern, modulated by the activity of the first layer. A response is measured in the form of some observable of the second layer's dynamics, and is dependent on the interplay between the input, the stabilized orbit, and the attractor itself. The application in [2] follows this design. As with versions 2. and 3. below, the spatiotemporal symmetries of the attractor, the input pattern, and/or the stabilized UPO, play an important role.

2. The dynamics of a single-layer network is driven into an UPO, by control or targeting methods. As soon as an external pattern starts to perturb the system, control is switched off. An output value or signal is measured from the transient dynamics of the system in response to the input pattern. This response depends on the initial dynamical condition resulting from momentarily having stabilized an adequate UPO in view of the task. This is the design we choose for the example below in this paper.

3. The dynamics of a single-layer network is not driven into any UPO, but a response to each input pattern is measured as in 2.

We have tested design 2. with the initial purpose (for this biologically inspired model) of assessing basic features such as a fading memory, the separation property [6], and the ability of at least performing standard digital computation. Such exploratory tests gave evidence of the mentioned properties, but they would be overly detailed to mention here. For brevity, we focus on the example of computing Boolean functions. The AND, OR and NOT functions could be easily computed, but here we illustrate only the computation of the XOR. The case of more complex (dynamical) input patterns is not addressed at present.

The system receives external input in the form of direct perturbations p_i of excitatory neurons X_i. Even if only one excitatory neuron is perturbed, the perturbation will spread through the entire network. An external perturbation of neuron X_i is numerically encoded by changing the resting potential V_L of X_i by some positive or negative amount. Such perturbations p_i can have any type of variation in the course of time and also across different spatial positions in the network. Here, however, we choose the p_i to either be zero or some small positive constant. To compute the function $\mathsf{XOR}\,(z_1, z_2)$ the following encoding is used: $p_i = 0.0$ for $i = 1, 3..6, 8$; $p_2 = 0.5$ if $z_1 = 1$ (or True), $p_2 = 0.0$ if $z_1 = 0$ (or False); z_2 is similarly encoded by p_7.

For this computational task, a totally symmetric orbit is initially targeted or stabilized. This is the period-one UPO corresponding to bulk oscillations of the network as described in [4]. An external perturbation, or pattern, is introduced at time 0.0. At this moment, the UPO stabilization procedure is turned off. The transient dynamical response of the network is then monitored.

A readout function is chosen in the form of the running standard deviation of the B_1 time-series, where B_1 is the coefficient of the first sine mode from a Fourier spatial projection of the network activity as defined in [4]. The sliding window for computing the standard deviation measures 46 msec. This output signal should be able to separate the four possible input patterns $(0, 0)$, $(0, 1)$, $(1, 0)$ and $(1, 1)$ into two different classes according to the well-known XOR truth table.

The network's response to each input pattern is shown in Fig. 2. The output signal clearly separates the two classes as desired already at time $t = 23$ msec, and this separation lasts until $t = 110$ msec. By $t = 200$ msec the time-series become harder to disentangle. The high symmetry of the orbit chosen for initial stabilization, along with the symmetry of the input encoding, are the cause for the responses to $(0, 1)$ and to $(1, 0)$ to practically overlap initially. With a moderate amount of noise, these two time-series would be decoupled, but the qualitative separation in response would be kept for the four input patterns (not shown in this experiment).

Assuming that some process is available for capturing the output signal of Fig. 2, no special consideration needs to be given to further readout neurons or the learning thereof. Class separation can be achieved trivially in a linear fashion, for the XOR as well as for the other Boolean functions mentioned above.

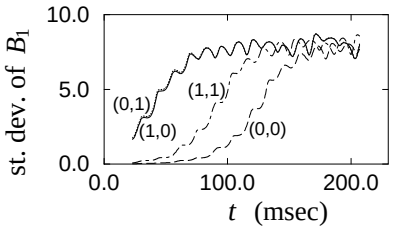

Fig. 2: Network spatial response signal as a function of the input pattern (see text for details of the output signal). This response allows to separate the four input patterns into two classes: $[(0,0) \rightarrow$ dashed line] and $[(1,1) \rightarrow$ dash-dotted line] in one class; $[(0,1) \rightarrow$ dotted line] and $[(1,0) \rightarrow$ solid line] in the other class.

3 Perspectives

Exploring a chaotic attractor's structure through its Unstable Periodic Orbits allows us to use a structured reservoir without sacrificing flexibility. The interplay between the spatiotemporal symmetries of dynamical input patterns and of internal states of the neural network plays an important role in this type of computing. A simple example was provided in this paper. The possibility of processing more complex patterns with these biologically inspired networks should be investigated. More elaborate input patterns, as well as different types and shapes of output signals, may justify including an additional set of simple neurons capable of learning. An investigation of truly dynamical patterns might start with the elementary pattern approach of [2], which already includes non-static input. A most relevant research topic is the purposeful design of chaotic attractors and respective UPOs, which may be instrumental in tailoring the neural network in regard to the required computational tasks.

References

[1] A. Babloyantz and C. Lourenço, Computation with chaos: A paradigm for cortical activity, *Proceedings of the National Academy of Sciences USA*, 91: 9027-9031, 1994.

[2] C. Lourenço, Attention-locked computation with chaotic neural nets, *International Journal of Bifurcation and Chaos*, 14:737-760, 2004.

[3] C. Lourenço, Dynamical reservoir properties as network effects. In M. Verleysen, editor, *proceedings of the* 14^{th} *European Symposium on Artificial Neural Networks* (ESANN 2006), d-side pub., pages 503-508, April 26-28, Bruges (Belgium), 2006.

[4] C. Lourenço, Dynamical computation reservoir emerging within a biological model network. To appear in *Neurocomputing*, 2007, doi:10.1016/j.neucom.2006.11.008

[5] H. Jaeger, The "echo state" approach to analyzing and training recurrent neural networks, *GMD Report* 148, German National Research Center for Information Technology, 2001.

[6] W. Maass, T. Natschläger and H. Markram, Real-time computing without stable states: A new framework for neural computation based on perturbations, *Neural Computation*, 14:2531-2560, 2002.

[7] D. Verstraeten, B. Schrauwen, M. D'Haene and D. Stroobandt, The unified Reservoir Computing concept and its digital hardware implementations. In *Proceedings of the 2006 EPFL LATSIS Symposium*, pages 139-140, 2006.