Toward optical signal processing using Photonic Reservoir Computing

Kristof Vandoorne,[1*] Wouter Dierckx,[1] Benjamin Schrauwen,[2] David Verstraeten,[2] Roel Baets,[1] Peter Bienstman,[1] and Jan Van Campenhout[2]

[1]*Photonics Research Group, Dept. of Information Technology, Ghent University – IMEC, Sint-Pietersnieuwstraat 41, 9000 Gent, Belgium*
[2]*PARIS, Dept. of Electronics and Information Systems, Ghent University, Sint-Pietersnieuwstraat 41, 9000 Gent, Belgium*
[*]*Corresponding author: Kristof.Vandoorne@UGent.be*

Abstract: We propose photonic reservoir computing as a new approach to optical signal processing in the context of large scale pattern recognition problems. Photonic reservoir computing is a photonic implementation of the recently proposed reservoir computing concept, where the dynamics of a network of nonlinear elements are exploited to perform general signal processing tasks. In our proposed photonic implementation, we employ a network of coupled Semiconductor Optical Amplifiers (SOA) as the basic building blocks for the reservoir. Although they differ in many key respects from traditional software-based hyperbolic tangent reservoirs, we show using simulations that such a photonic reservoir can outperform traditional reservoirs on a benchmark classification task. Moreover, a photonic implementation offers the promise of massively parallel information processing with low power and high speed.

OCIS codes: (190.4390) Nonlinear Optics, integrated optics; (250.5980) Semiconductor Optical Amplifiers; (200.4700) Optical neural systems

References and links

1. H. Jaeger and H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication," Science **304,** 78–80 (2004).
2. W. Maass, T. Natschlager and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," Neural Computing **14,** 2531–2560 (2002).
3. W. Maass, T. Natschläger, H. Markram, "A model for real-time computation in generic neural microcircuits," in *Proceedings of NIPS,* (MIT Press, Vancouver, British Columbia, 2003), pp. 229–236.
4. M. D. Skowronski, J. G. Harris, "Minimum mean squared error time series classification using an echo state network prediction model," in *Proceedings of IEEE International symposium on circuits and systems* (Institute of Electrical and Electronics Engineers, Island of Kos, Greece, 2006).
5. D. Verstraeten, B. Schrauwen, D. Stroobandt, and J. Van Campenhout, "Isolated word recognition with the Liquid State Machine: a case study," Information Processing Lett. **95,** 521–528 (2005).
6. H. Jaeger, "Reservoir riddles: Suggestions for echo state network research (extended abstract)." in *Proceedings of IEEE International Joint Conference on Neural Networks* (Institute of Electrical and Electronics Engineers, Montreal, 2005), pp. 1460–1462.
7. P. Joshi, W. Maass, "Movement generation and control with generic neural micrrocircuits," in *Proceedings of BIO-ADIT* (2004), pp. 16-31.
8. J. J. Steil, "Online stability of backpropagation-decorrelation recurrent learning," Neurocomputing, **69,** 642–650 (2006).
9. H. Y. S. Li, Y. Qiao, and D. Psaltis, "Optical Network for Real-Time Face Recognition," Appl. Opt. **32,** 5026-5035 (1993).

#95957 - $15.00 USD Received 8 May 2008; revised 27 Jun 2008; accepted 9 Jul 2008; published 10 Jul 2008

10. B. Javidi, J. Li, and Q. Tang, "Optical Implementation of Neural Networks for Face Recognition by the Use of Nonlinear Joint Transform Correlators," Appl. Opt. **34,** 3950–3962 (1995).
11. M. Hill, E. Edward, E. Frietman, H. de Waardt, H. J. S. Dorren, and G. Khoe, "All Fiber-Optic Neural Network Using Coupled SOA Based Ring Lasers," IEEE Trans. Neural Networks, **13,** 1504–1513 (2002).
12. C. M. Bishop, *Neural Networks for Pattern Recognition* (Clarendon Press, Oxford, 1995).
13. V. N. Vapnik, "An overview of statistical learning theory," IEEE Trans. Neural Networks **10,** 988–999 (1999).
14. R. Legenstein and W. Maass, "What makes a dynamical system computationally powerful?" in *New directions in statistical signal processing : from systems to brain*, S. Haykin, ed. (MIT Press, Cambridge, MA, 2007).
15. Reservoir lab, "Reservoir Computing Toolbox". http://www.elis.ugent.be/rct
16. D. Verstraeten, B. Schrauwen, M. D'Haene, and D. Stroobandt, "An experimental unification of reservoir computing methods," Neural Networks **20,** 391–403 (2007).
17. G. P. Agrawal and N. A. Olsson, "Self-Phase Modulation and Spectral Broadening of Optical Pulses in Semiconductor-Laser Amplifiers," IEEE J. Quantum Electron. **25,** 2297–2306 (1989).
18. H. S. Rong, Y. H. Kuo, S. B. Xu, A. S. Liu, R. Jones, and M. Paniccia, "Monolithic integrated Raman silicon laser," Opt. Express **14,** 6705–6712 (2006), http://www.opticsinfobase.org/abstract.cfm?URI=oe-14-15-6705.
19. M. Cernansky and M. Makula, "Feed-forward echo state networks," in *Proceedings of IEEE International Joint Conference on Neural Networks* (Institute of Electrical and Electronics Engineers, Montreal, 2005), pp. 1479–1482 vol. 1473.
20. A. N. Tikhonov and V. I. Arsenin, *Solutions of ill-posed problems* (Winston & Sons, Washington, 1977).
21. H. Jaeger, "Adaptive nonlinear system identification with echo state networks," in *Proceedings of NIPS*, (MIT Press, Cambridge, MA, 2003), pp. 593–600.

1. Introduction

We propose photonic reservoir computing as a new framework to handle classification and regression problems. It is based on a photonics implementation of the recently proposed reservoir computing concept [1, 2], where the internal dynamics of a large network of nonlinear elements are used to distinguish between different time-varying signals, and in this way perform a classification task.

The traditional incarnation of reservoir computing is a purely software based one, where the nonlinearities are often the *tanh*–based nodes found in neural networks. Such reservoirs have been employed successfully in a large variety of applications like speech recognition [3–5], event detection [6], robot control [7] and chaotic time series generation and prediction [1, 8].

Rather than simulating a nonlinear element using a software algorithm, we propose to implement such an element using a photonics device. This could have advantages in terms of speed and power efficiency. Moreover, due to the interplay between carriers and photons, photonic nodes can have internal dynamics which are much richer than the static *tanh* nodes used in software nodes, which could result in better processing capabilities. However, it is not entirely obvious whether simply translating the *tanh* reservoir concept into photonics would be successful, as there are fundamental differences between a software *tanh* reservoir and a photonics reservoir. Firstly, light levels in a photonics implementation are always positive, whereas excitation levels in a neural network can be both positive and negative. Secondly, there are no practical limitations on the interconnection topology in a software implementation, whereas in e.g. a nano-photonic IC based implementation, many crossing interconnections between the elements are to be avoided for technological reasons.

In this paper, we will use numerical simulations to show that despite these fundamental differences, a reservoir of coupled SOA's can perform very well on a non-trivial classification task, even sometimes outperforming software-based *tanh* reservoirs because of the richer internal dynamics in the photonic nonlinear elements.

This paper is structured as follows. Because we assume that many readers from the photonics community are not familiar with recent advances in the field of machine learning, we will go deeper into the concept of reservoir computing in section 2. In section 3, we will discuss in more detail the photonic implementation aspects. Section 4 describes the simulation model we employed to assess the behavior a large network of coupled SOA's. Subsequently, in sec-

#95957 - $15.00 USD Received 8 May 2008; revised 27 Jun 2008; accepted 9 Jul 2008; published 10 Jul 2008

tion 5 we show how the photonic reservoir performs in a non-trivial classification task, namely distinguishing between a triangular and a rectangular waveform. It turns out that a photonic reservoir — with only minimal tuning and a limited number of 25 SOA's — can already distinguish between the two signals over 97% of the time, which is better than a software based *tanh* implementation. Finally, in section 6 we give some conclusions and present a broader outlook.

2. Software based reservoir computing

In this digital age, signals are often transformed into the digital domain for processing. Nature, however, shows us that there are alternative methods of information processing, which can be superior for complex classification and regression problems. One example is the performance of the human brain when it comes to recognizing patterns in a complex visual scene. The field of machine learning looks at the biological world as a source of inspiration for building classification systems. Neural networks are an example of such systems, consisting of large numbers of nonlinear nodes (the neurons), which are interconnected among each other using links with a certain weight. It is precisely these weights which are adapted during the training of the neural network in order to realize a system with a certain desired behavior. Several optical implementations of neural networks have already been made in the past [9–11] .

Feed-forward neural networks are structured in different layers where information only flows from one layer to the next. This corresponds to a time-invariant nonlinear mapping from the input to the output [12]. They have been extensively used for non-temporal problems and they are well understood due to their non-dynamic nature. At the same time this limits their applicability in dealing with time varying signals. Indeed, neural networks with feedback loops (so-called recurrent neural networks) provide some kind of internal memory which allows them to extract time correlations. However, such recurrent neural networks turn out to be very difficult to train, which limits their broad applicability.

Around 2002, two groups independently came up with a solution for this problem [1, 2]. The philosophy is to split the classifier in two parts. One part — the so called reservoir — is a random recurrent neural network that is left untrained and kept fixed during use. The time-varying input is fed into this reservoir and gives rise to complex internal dynamics. The second part of the classifier is a readout function, which takes as input the instantaneous reservoir state, i.e. the collection of all of the states of the individual elements, and produces an output decision. In order to be able to achieve useful functionality, this part of the system needs to be trained, typically on a set of inputs with known classifications. This process is visualized in Fig. 1, where the bars represent vectors in case of a multi-dimensional input, state or output. Any kind of classifier or regression function could be used, but it turns out that for most applications a simple linear regression suffices. In this way the interesting properties of recurrent neural networks are kept in the reservoir part, while the training is now restricted to the least-squares optimization of a linear memory-less readout function.

One might wonder why such an approach would be useful to solve complex classification tasks. However, it is well known in the machine learning community that projecting a low-dimensional input into a high-dimensional space can actually be beneficial for the performance of a classification algorithm, as classes which are separable only by a complicated nonlinear surface in the low-dimensional space, can become separable by a linear hyperplane in the high-dimensional space. This concept is the basis of a broad range of machine learning methods, called Kernel methods [13].

The reservoir in a way converts the temporal correlations present in the signal into spatial correlations in the reservoir state. This is not to say that any recurrent neural network will achieve this. Rather, it appears that the dynamics of the network should ideally lie on the edge of stability [14]. This is usually achieved by tuning the amount of gain and loss in the network. If

#95957 - $15.00 USD Received 8 May 2008; revised 27 Jun 2008; accepted 9 Jul 2008; published 10 Jul 2008

the network is over-damped, then there is no memory inside the reservoir; if it is under-damped then the network will react too chaotically.

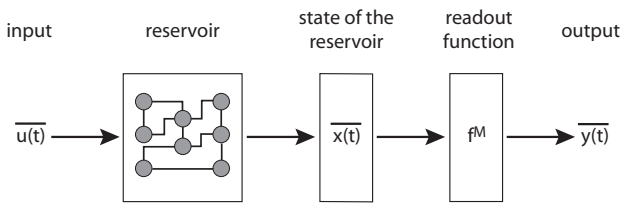

Fig. 1. Reservoir Computing

The reservoir used in reservoir computing does not necessarily have to be a neural network. Indeed, there have been conceptual implementations of reservoir computing using the surface of a real liquid as the reservoir, with objects being dropped into the liquid providing the time-varying inputs which set up complicated dynamical patterns on the liquid surface. There are, however, certain non-restrictive requirements that a complex nonlinear system needs to meet in order to be suitable as a computing reservoir. One of them is *fading memory*, which means that the influence of past inputs slowly fades away, and the network state asymptotically becomes independent of the initial conditions. Apart from these very broad requirements, the experience teaches us that the performance of the reservoir is fairly robust in terms of variations of the exact details of the nodes. Indeed, software reservoirs are mostly created randomly, i.e. with a random interconnection topology, and then tuned globally by rescaling the gain and input levels so their dynamics approximately match the edge of stability. This robustness of reservoir computing under perturbations of the reservoir is in strong contrast with digital electronics, where a single bit error can have disastrous consequences.

Recently a toolbox was created which is able to simulate and test a wide variety of reservoirs, mostly based on neural networks [15]. One of those is the classical variant where the signals are analog and every node is a hyperbolic tangent function operating on a time-integrated weighed sum of its inputs. This function is S-shaped as in Fig. 2 (left). In this kind of network the nodes themselves are very simple, while the dynamics arises from the complex interconnection topology. It is interesting to keep this in mind when comparing this to our proposed photonic implementation of reservoir computing.

3. Photonic reservoir computing

The present software implementations of reservoir computing are rather slow (typically 1 ms per simulation timestep) and therefore we investigate the potential of a hardware implementation based on light. This could have advantages in terms of speed and power efficiency due to the large bandwidth and fast nonlinear effects that are possible in a photonic implementation.

Because of the nature of reservoir computing, its implementation can be split up in two distinctive parts: the reservoir and the readout function. Since the computational power of reservoir computing resides mainly in the reservoir due to its feedback and nonlinearities, the focus of the current research is on a photonic reservoir, coupled to an off-line electronic readout. As mentioned before, the readout is most often a simple linear discriminant, but its training depends on mathematical calculations like matrix pseudo-inversion. This could initially be done off-line by a traditional computer or a dedicated electronic chip.

Since the requirements on the nature and the characteristics of the reservoir do not seem to be very stringent, we are faced with a vast choice of nonlinear components and effects to try and realize the reservoir. For the current study, we do not aim to design a system with optimal

#95957 - $15.00 USD Received 8 May 2008; revised 27 Jun 2008; accepted 9 Jul 2008; published 10 Jul 2008

performance in terms of speed and power usage, but rather we investigate a photonic reservoir which easily lends itself to future experimental verification.

We opted for a photonic chip with a network of many coupled Semiconductor Optical Amplifiers (SOA). We made this choice based on several observations. First of all, the steady-state power transfer curve of an SOA resembles the upper branch of the *tanh*-curve used in analog neural networks (Fig. 2), and thus seems an obvious choice for a first implementation of a photonic reservoir. SOA's also lend themselves to compact integration on a photonic integrated circuit. Although the nonlinearities in SOA's are not extremely fast, they are broadband which makes the communication between adjacent nodes in the reservoir rather straightforward. A reservoir based on resonant photonic crystal cavities offers the potential for much faster nonlinear effects, but the potential mismatch in resonance wavelength between adjacent nodes could mean that signals cannot propagate very far into the network. This makes them less suited for a first prototype.

We stress the fact that it is not immediately obvious whether such a network of SOA's will make a good reservoir. Compared to *tanh* reservoirs, SOA's lack the symmetric lower branch of the *tanh* because optical power is non-negative. Also, if we want to implement this network on a chip, we are limited to non-crossing interconnections, which is inherently less rich than the random interconnection in software reservoirs. In spite of these two limitations, the SOA however has richer *internal* dynamical behavior as opposed to the static neurons used in *tanh* software reservoirs, which could be computationally more powerful. This dynamic behavior comes into play at higher data rates because of the interaction between the photons and the carriers. Since it is not a priori clear how the performance of this photonic reservoir will compare with a traditional *tanh* reservoir, we need to perform detailed simulations to resolve this issue. Answering this question is the main contribution of this paper.

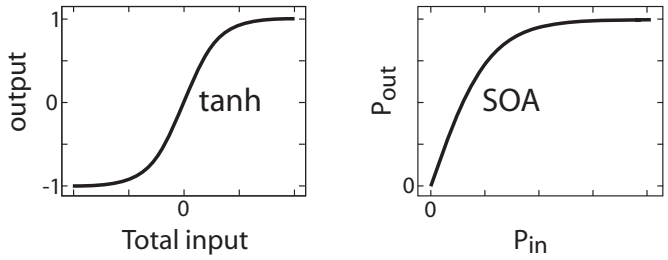

Fig. 2. (left) *tanh* transfer characteristic used in analog neural network — (right) SOA: steady state power transfer curve

4. Simulation model

We developed our simulation program for photonic reservoirs within the framework of the toolbox mentioned previously. This allowed us to utilize the existing training and evaluation schemes for the memory-less readout function. For further details of this open source toolbox we refer to the online manual [15] and the paper by D. Verstraeten et al. [16] where the toolbox was first introduced.

4.1. SOA model

In our simulations we work with traveling wave SOA's. This kind of SOA has anti-reflection coatings on its facets, which allows us to neglect the influence of reflections. We use the standard traveling wave SOA equations [17]. $P(z,\tau)$ and $\phi(z,\tau)$ represent the power and phase and

#95957 - $15.00 USD Received 8 May 2008; revised 27 Jun 2008; accepted 9 Jul 2008; published 10 Jul 2008

when the internal losses are neglected, they can be calculated through equations 1 and 2:

$$P_{out}(\tau) = P_{in}\exp[h(\tau)] \quad (1)$$

$$\phi_{out}(\tau) = \phi_{in} - \frac{1}{2}\alpha h(\tau), \quad (2)$$

where α is the linewidth enhancement factor and the reduced time $\tau = t - z/v_g$ is measured in a reference frame moving with the light. The function $h(\tau)$ is the gain $g(z,\tau)$ integrated over the length L of an SOA (equation 3):

$$h(\tau) = \int_0^L g(z,\tau)dz. \quad (3)$$

Its value can be calculated by the following ordinary differential equation:

$$\frac{dh}{d\tau} = \frac{g_0 L - h}{\tau_c} - \frac{P_{in}(\tau)}{P_{sat}\tau_c}[\exp(h) - 1], \quad (4)$$

where τ_c is the spontaneous carrier lifetime, g_0 the small signal gain and P_{sat} the saturation power of the amplifier. For the theory behind these equations we refer to the work by Agrawal et al. [17]. The values we typically use are $\alpha = 5$, $n_g = 3.75$, $L = 500\,\mu$m, $\tau_c = 300$ ps, $g_0 = 6075\,\text{m}^{-1}$ and $P_{sat} = 0.0211$ W. In this way the model approximates the steady-state curve of a physically realized IC SOA. We apply an extra loss of 6 dB to account for the internal losses.

To incorporate the longitudinal dependence of the gain, the equations can be solved for a concatenation of small sections of the SOA. Since the latter can be time consuming when working with large networks of SOA's, we work mainly with one section. Moreover, since reflections are neglected at this stage, we use unidirectional signal injection. This reduces the number of rate equations to be solved to one.

We neglect the influence of Amplified Spontaneous Emission (ASE) and the wavelength dependence of the semiconductor gain in this model. This means that we assume that the input signal itself will be strong enough to dominate the ASE and that we will only use light at one wavelength — 1.55 μm in our experiments. To assess the robustness of the reservoir with respect to phase effects, we simulated the reservoir with different wavelengths and up to a 100 MHz frequency difference relative to 1.55 μm, the behavior did not change noticeably. This is well in reach of some lasers [18].

4.2. Topology and reservoir simulation

The classical reservoir implementations with neural networks have random interconnection topologies. Since on a 2D optical chip waveguide crossings are best avoided, we investigated structures that can be realized without intersections. Two of those structures are depicted in Fig. 3. The left structure is a waterfall system which acts as a nonlinear delay-line. Although this feed-forward topology is relatively simple, it has already been successfully used to model nonlinear systems [19]. The other network has feedback connections on the sides in order to avoid crossings.

Every node in the network is modeled as an input–output system and therefore all the connections are also unidirectional. At every time instant two computational steps are taken. During the first step the internal state of every node is updated, while during the second step the outputs are transferred to the inputs they are connected to. The splitters and combiners are modeled as adiabatic and we kept the weights fixed for every node. Every connection can have a different delay and attenuation. The delays we used are compatible with delays on chips.

#95957 - $15.00 USD Received 8 May 2008; revised 27 Jun 2008; accepted 9 Jul 2008; published 10 Jul 2008

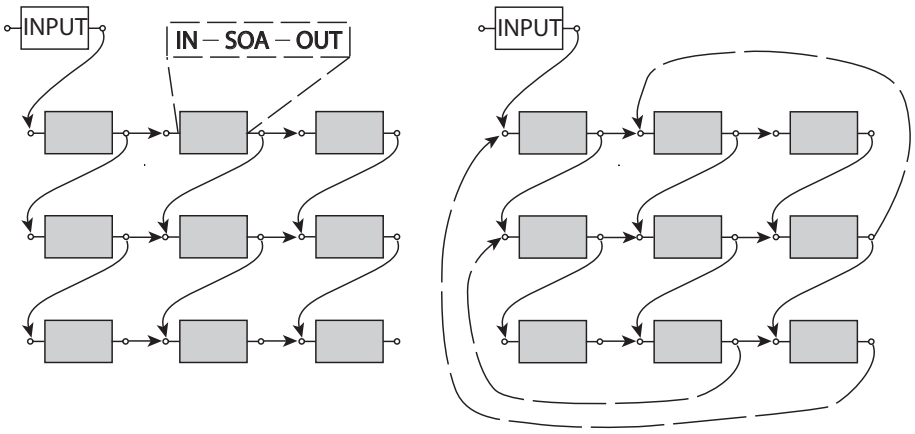

Fig. 3. Two topologies: (left) a feed-forward network – (right) a waterfall network with feedback connections (long dash) at the edges

4.3. Readout function

The readout function is a memory-less linear combination of the instantaneous states of all the nodes, with weights which are determined during training. This is accomplished by minimizing $||Aw - B||^2$ where A is a matrix holding all the node states, w is the weight matrix and B consists of all the desired outputs. The weights w can be found using the Moore-Penrose pseudo inverse. One problem with this solution is that it tends to overfit the training data and will perform worse on unseen data. A solution is to control the model complexity which can be accomplished by keeping the norm of the weights small. To keep the weights small, an additional penalty term, proportional to the weight's norm, is added as can be seen in equation 5:

$$||Aw - B||^2 + \lambda \, ||w||^2 \tag{5}$$

The value of the regularization parameter λ determines to what extent large weights are penalized as it keeps the norm of the weight matrix small. The weights themselves can be found in a single step using ridge regression, also called Tikhonov regularization [20]. The optimal value for λ is determined through cross-validation on a distinct validation set.

This is the basic structure of the simulation model. Next we will look at tasks that can be solved with these kind of networks.

5. Pattern recognition benchmark

5.1. Task setup

We will use a simple but non-trivial classification task as a first benchmark to demonstrate the potential of photonic reservoir computing. An example of this task is depicted in Fig. 4, which shows a result of an SOA reservoir with the topology shown in Fig. 3(b) and a 2 dB node-to-node attenuation. This corresponds to the optimum of the blue curve in Fig. 5 for a reservoir with 25 SOA's and 6.25 ps delays between them. Through training by examples, the system has to be able to instantly distinguish between a rectangular and a triangular waveform. Moreover, if the input signal changes the system has to change its output as fast as possible. In Fig. 4(a), an example of such an input is depicted. Figure 4(b) shows the output that the system should generate: if the input is triangular, the output should be 1, if the input is rectangular it should return -1. Figure 4(c) shows the state (i.e. their optical power level) of a few SOA's — chosen for illustrative purposes out of the network of 25 SOA's — as they are excited by the input,

#95957 - $15.00 USD Received 8 May 2008; revised 27 Jun 2008; accepted 9 Jul 2008; published 10 Jul 2008

while Fig. 4(d) shows the result of the readout function. Note that the weights used by the off-line readout function can be negative and in such a way negative outputs can be realized. (As mentioned before, we envisage this training and readout in a first stage to be done off-line by an electronic chip). Figure 4(e) shows the final output of the system, obtained by applying a sign function on the result of the linear combination. In the example the system manages most of the time to generate the desired output, barring some discrete spikes.

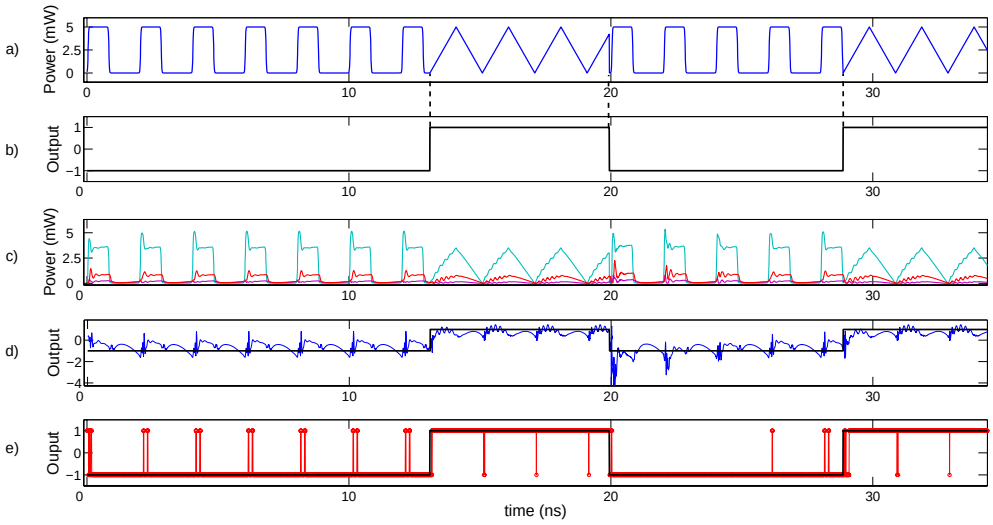

Fig. 4. Pattern recognition task: a) Input signal with different transitions between the rectangular and triangular waveform b) desired output c) state (i.e. optical power level) of some of the reservoir nodes d) The approximation (blue) of the desired output (black) by the readout function, e) final output of the system (red)

Figure 4 shows only qualitatively how the reservoir performs. We will now proceed to give more a quantitative evaluation of the behavior of the reservoir.

5.2. Error rate

We now calculate the error rate of different photonic reservoirs on the classification task described earlier. For this, we proceed as follows. Choosing a certain reservoir, we first train the weights of the readout function using an input signal of length 100 ns, together with its desired output. Afterwards, the weights are kept fixed, and a new, different input signal of length 100 ns is presented to the reservoir. The output of the system is compared to the correct desired output, and an error rate is defined as the percentage of time that the reservoir gives an incorrect answer.

In Fig. 5(a), we plot the error rate for the two photonic topologies from Fig. 3, with 25 SOA's, as a function of the globally set attenuation in the connections.

This is an important parameter, as it allows us to tune the system to the dynamic regime at the edge of stability, where the computational power is the largest. In a practical implementation, instead of tuning the loss in the connections, we can obtain a similar effect by tuning the pump level of the SOA's.

The best result is obtained using feedback in the system and corresponds to an error rate of 2.5%. We see that if the attenuation in the system gets too small, the error rate increases dramatically for the feedback network, as its dynamics become too chaotic.

#95957 - $15.00 USD Received 8 May 2008; revised 27 Jun 2008; accepted 9 Jul 2008; published 10 Jul 2008

The vertical lines are error bars, which show the standard deviation on the reservoir performance over ten runs. The variation comes, for the photonic reservoirs, from different input signals with different transitions at different instants. The *tanh* networks have an extra variation source because they are randomly created.

In Fig. 5(b), we compare a feedback reservoir with SOA's to the classical *tanh* reservoirs, each with 25 nodes. The classical *tanh* reservoirs have a random interconnection topology with random Gaussian distributed weights, as opposed to the SOA–feedback network which has the rather symmetric topology of Fig. 3(b) and the same *positive* weights for all its connections. This time we use the spectral radius $(\rho(\cdot))$ as a measure for the dynamics in the system (so the curve for the SOA network with feedback is the same in fig.5a and 5b, but plotted against a different parameter.) The spectral radius is the largest absolute eigenvalue of the connection matrix *C*, containing all the gain and loss in the network and is an often used parameter in the field of reservoir computing. In a linear network, a spectral radius smaller than one means the network is stable, a value larger than one means thats its state will go to infinity. For the nonlinear *tanh* reservoir which is locally linear around the origin, a spectral radius larger one than will lead to a wildly oscillating, locked up or even chaotic system. The interesting dynamical region, the edge of stability, holds for spectral radii just below one. Although our network is nonlinear, we can still use this as an approximation, where the spectral radius acts as an upper estimate, since smaller input powers experience the strongest gain in an SOA. In the SOA network the connection matrix C contains the total gain and loss from every node to every other node. The total loss is determined by the weights of the in- and output couplers of every SOA and the losses in the waveguides in between the SOA's; as for the gain of every SOA, the value obtained when linearizing its gain around zero power, where it is highest, is used. The time delays are neglected in this calculation. For a connection matrix *C* with eigenvalues $\lambda_i, \ldots, \lambda_n$ this leads to the following spectral radius calculation:

$$\rho(C_{lin(\langle 0 \rangle)}) = \max_{1 \leq i \leq n} |\lambda_i| \tag{6}$$

The *tanh* reservoir appears to behave better for small spectral radii, which corresponds to a more linear operation of the nodes, with an optimum error rate around 3.5%. As soon as the spectral radius gets too high, the system becomes chaotic which explains the large error bars and poor mean performance. The optimal regime for these tanh reservoirs seems to extend from the edge of chaos into the stable regime for this task. The same holds for the photonic reservoirs but they have a clear minimum error value of only 2.5%, and this is obtained for a spectral radius around 0.5.

It is remarkable that the photonic reservoir with feedback is slightly better than the *tanh* reservoir for this task, considering the simple photonic topology used. One explanation for this is the different response of an SOA to different rise times. The rectangular waveform rises faster than the triangular one, causing a depletion of the carriers in the SOA. This can be seen in Fig. 4(c), where peaks appear whenever the rising edge of the rectangular waveform passes through an SOA.

This result indicates that the poorer connectivity and lack of a symmetric lower branch in the transfer characteristics are more than compensated by the richer internal dynamics of the nodes.

Finally, in Fig. 6 we plot the error rate as a function of reservoir size. We see that the performance improves with larger reservoirs, although there is no longer a significant improvement beyond 25 nodes.

We also tested the photonic reservoir on two other benchmark tasks. One task evaluates the memory capacity of the network, the other was the identification of a nonlinear ARMA task [21]. Here too, the results are promising and will be published elsewhere.

#95957 - $15.00 USD Received 8 May 2008; revised 27 Jun 2008; accepted 9 Jul 2008; published 10 Jul 2008

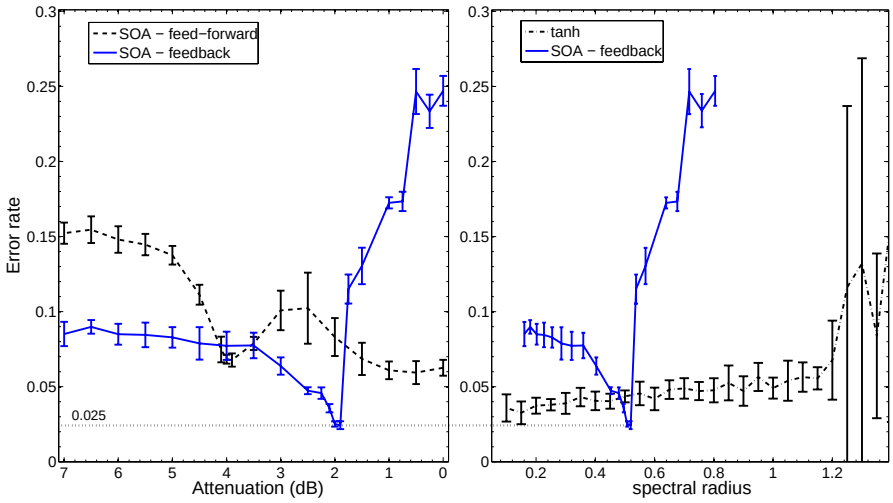

Fig. 5. Results with a signal frequency of 0.5 GHz, simulation time: 100 ns, amplitude power: 5 mW, delay: 6.25 ps, 25 nodes: (left) – photonic reservoirs with and without feedback, (right) – classical reservoirs versus photonic reservoirs with feedback

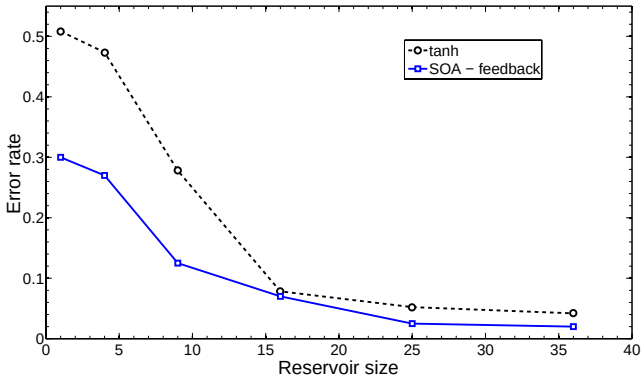

Fig. 6. This figure shows the influence of the reservoir size on the performance of the classical and photonic reservoirs with feedback.

#95957 - $15.00 USD Received 8 May 2008; revised 27 Jun 2008; accepted 9 Jul 2008; published 10 Jul 2008

6. Conclusions and outlook

We proposed photonic reservoir computing as a novel approach to do optical signal processing. We have seen that despite several limitations compared to traditional *tanh* reservoirs (lack of negative excitations, poorer interconnectivity), the richer internal dynamics of SOA-based nodes can result in performance which for some tasks is even superior as compared to these of *tanh* reservoirs. This is a promising step toward the use of photonic reservoirs for large scale and high speed classification problems.

Further research is concentrating on two different fronts. First of all, we are working toward an experimental verification of the results presented in this paper. Second, we are investigating other types of applications which could benefit from photonic reservoir computing: speech recognition, large scale image recognition in real-time video data, header extraction in optical data signals, ...

Acknowledgments

KV acknowledges the Special Research Fund of Ghent University (BOF – UGent) for a doctoral mandate. This work was performed in the context of the Belgian IAP project Photonics@BE.

#95957 - $15.00 USD Received 8 May 2008; revised 27 Jun 2008; accepted 9 Jul 2008; published 10 Jul 2008