Brain inspired electro-optical neuro-computing: from ideas to real implementations

Sergei Rybalko *Romain Martinenghi* *Lennert Appeltant* *Guy Van der Sand*
Jan Danckaert *Maxime Jacquot* *Yanne Chembo* *Laurent Larger*

1 Liquid State Machine as a complex dynamical system

1.1 Basic principles of Reservoir Computing

Reservoir computing has recently emerged as the generic name of a new research field in machine learning, which combines Echo State Networks (ESN), Liquid State Machines (LSM) and Backpropagation Decorrelation algorithms (BPDC). These systems that can be used to solve complex classification problems have specific properties in common and are considered a new computational paradigm for neural networks. Reservoir computing is based on recurrent activity, so far of either many coupled analogue elements or many pulse-coupled spiking elements like neurons [1, 2, 3, 4]. Computations emerge from the properties of the individual coupled elements and the inherent network dynamics. Therefore, reservoir computing links temporal and spatial processing based on computations on evolving trajectories in a high dimensional state space of the system. The reservoir usually consists of a set of interconnected nonlinear dynamical elements in a given (fixed) network. The dynamical response of the reservoir is determined, by both the input drive and the past activity of the reservoir (Fig.1).

Reservoir Computing (RC) or Liquid State Machine (LSM) is novel paradigm in neural networks. It consist of three main components: input layer, reservoir and output layer. All elements of network formed by elementary neural nodes. The main difference between conventional neural network and LSM is that besides it contains many elements (from several hundred to thousands) the only connection which could be trained are connections to output layer. It means that LSM can be trained very fast and perform many tasks of conventional neural networks with same efficiency and sometime even better. Clue principle of working of LSM is that it uses most of its nodes (reservoir layer) as high-dimensional dynamical system which react on input signal and translate it in higher dimension then input signal itself increasing separability (identification tasks) and due to non-linearity enhance its features (recognition tasks). LSM should has three main necessary features,

1. it should be high-dimensional dynamical system to transform close input signals in higher dimension and increase separability;
2. it should have stable steady state, without signals reservoir goes quickly to rest unexcited state;
3. it should be essentially nonlinear, separated input patterns should be far from each other in high dimensional space.

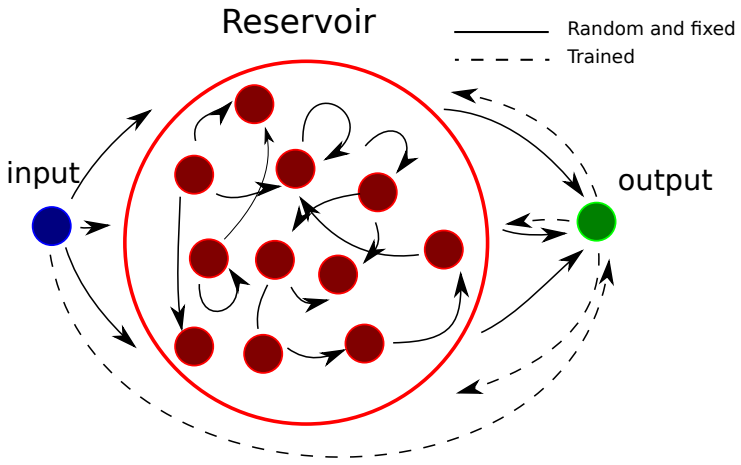

Fig. 1: Schematic representation of LSM

Numerically it was proven that a LSM can perform learning tasks better than conventional neural network ([1, 3] cite, Jaeger, Mass). Moreover it was shown that the performance depends besides many technical parameters on type of nodes (leaky neurons, spiking neurons etc.).

There are common steps in construction and evaluation of a reservoir system. First, one need to construct LSM which fulfill all criteria mentioned above. Second, a common task from machine learning should be selected to best fit to architecture of the build reservoir. It also means a choice of appropriate dataset. Then, training and validation should be performed.

To validate the performance of constructed reservoir all samples from chosen dataset go through cross-validation process. The hole set of samples is divided on n_p equal and non-intersecting subsets randomly, where n_p is integer number which can be called as *partition size*. After training and evaluation one can collect errors and continue cross-validation procedure with other arrangement of the partition. Next step is to take another subset (from n_p previously generated) as the test subset and the rest of the partition as the train pool. This procedure should be done for every subset from n_p subsets to be chosen as test one. It guarantees that every digit will be chosen only once as test digit.

Training process contains several steps. First one should collect all train samples and corresponding reservoir states and input node states into big matrix A. Main step of training is to find best read-out matrix W in such way that,

$$A \times W \simeq B,$$

where B is matrix of corresponding targets.

There are several techniques to solve this equation. The most common case, and the case we use in all experiments described here, is the ridge-regression,

$$W_{opt} = \arg\min_{W} \|AW - B\|^2 + \lambda \|W\|^2 .$$

Here, both norms are the standard Euclidean norms. The regression is known as ridge-regression or (less commonly) Tikhonov regression (Tikhonov and Arsenin, 1977).

While ridge regression offers good control over the smoothness of the obtained regression/readout function through a single parameter λ, the best value to use still needs to be determined. The optimal value for this regularization parameter would be the one that gives the best performance on unseen data. Determining this optimal value is done using grid-searching of the parameter with cross-validation.

After training the matrix W_{opt} is calculated. Now one should apply this matrix to test samples and compare the read-outs with targets. Evaluation measure depends on specific task.

1.2 Electro-optical setups as real physical implementation of LSM

From above introduction it is clear that LSM can be constructed as high-dimensional dynamical system, for example, spatio-temporal system. From the other side a dynamical system with delay is well-known example of infinite dimensional (in principles) system where a 'space' states are translated in to temporal 'state' due to delay line. The dynamics of delay system even can be represented by spatio-temporal picture where space coordinate is snap-shot system trajectory during one delay time (cite, from Romain paper[12]). It proves that delay system can play role of reservoir in hypothetical constructed LSM. In the rest of the paper we will theoretically and experimental demonstrate that a system with delay can be effectively used as LSM. Moreover because of systems with delay can be easily implemented as electro-optical laboratory setups ([13]cite Laurent and co) it opens perspectives to construct analog devices which can process information very fast and perform some neural network task with high accuracy. In the next parts of our paper we show it on two common neural network tests using two different but common in principles electro-optical setups.

2 Multiple delays photonic system

2.1 Setup

The setup we would like to present was used initially in the context of chaotic cryptography in wavelength dedicated optical telecommunication [13]. Its block diagram is given in Fig.2

It consists of a DBR diode laser emitting beam at 1550nm wavelength, a bandpass filter whose high and low time constants are θ and τ respectively and a voltage-to-current converter which controls the laser. Unlike the conventional electro-optical nonlinear generators with delay line, the delay line τ_D in this setup is produced digitally by programmable device. This delay is obtained via a logic circuit based on FPGA (*Field Programmable Gate Array*). FPGA allows to use its FIFO (*First In First Out*) coded in Very high speed integrated circuit Hardware Description Language (VHDL) in such way that the delay is adjustable by controlling the depth and the frequency of synchronization of FIFO memories.

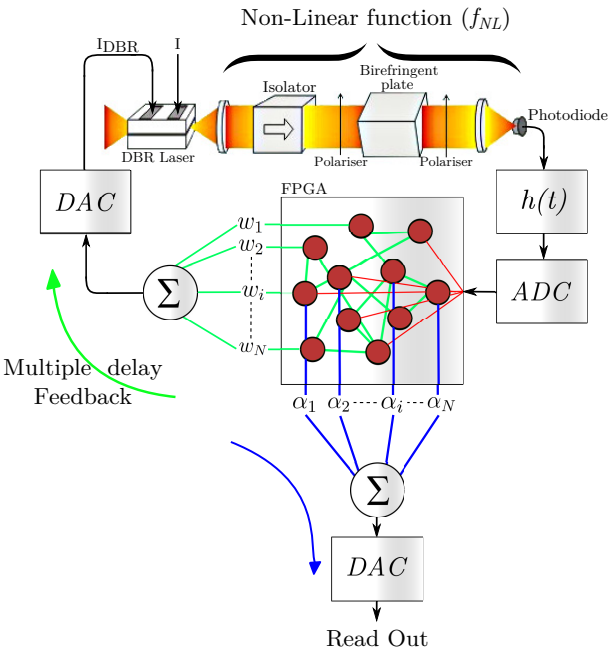

Fig. 2: Setup structure with multiple delay lines due to FPGA

	Frequential Domain	Temporal Domain
High pass filter	$2Hz < \theta^{-1} < 3100Hz$	$80ms < \theta < 51.84\mu s$
Low pass filter	$1200Hz < \tau^{-1} < 30kHz$	$132.629\mu s < \tau < 5.3\mu s$
Delay τ_D	$50kHz < \tau_D^{-1} < 100MHz$	$256.12ms < \tau_D < 128.06\mu s$

Tab. 1: Summary of the time constants of Reservoir Computing

Read-out from each FIFO memories is delayed by a period equivalent to the total delay τ_D divided by the number of nodes of reservoir N_{neur}. It allows to generate the 400 stimuli with amplitudes x_i coherent with the input node. Moreover FPGA board makes it is possible to apply a synaptic weight w_i (which are randomly generated beforehand) to each stimuli x_i using N_{neur} multipliers and finally to integrate all information before injection to nonlinear part. It is important to note that each node correspond to an elementary delay. The nonlinear dynamics is composed by ensemble of N_{neur} elementary delays and we can call setup as multiple delays dynamical system.

Experimental range of the adjustable time constants of high-pass filter and low pass filter and total delay τ_D are summarized Table1. The value of the parameters β in order to obtain the correct operation range of this setup are determined numerically.

2.2 Model

The mathematical formulation for given setup can be easily derived from basic principles. It is an electro-optical delay system with a multiple delays feedback loop. Thus the system can be described by an integro-differential equation with number of delays,

$$\tau\frac{\mathrm{d}x}{\mathrm{d}t} + x(t) + \frac{1}{\theta}\int x(s)\mathrm{d}s = \beta\sin^2\left[\sum_{i=1}^{N_{neur}} w_i x(t - i\cdot\delta\tau_D) + \gamma v(t)\varepsilon_n(t) + \phi_0\right]. \quad (1)$$

For simplicity of the numerical analysis the system were rewritten in form normalized to the response time of the low-pass filter. In above equation one should put $\tau = 1$.

In real experiments with FPGA setup, the number of neurons is $N_{neur} = 150$. It is caused by limitation on number of signal which FPGA can process simultaneously (number of FIFOs). As mentioned above weights w_i are random numbers normalized in such way that $\sum_{i=1}^{N_{neur}} w_i = 1$.

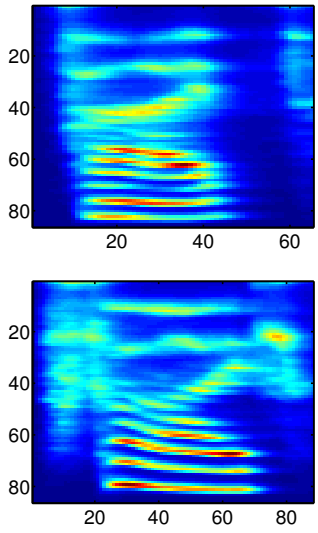

Fig. 3: Examples of same word 'six' uttered by different persons.

3 Spoken Digits Recognition task

3.1 Task formulation

Main goal of spoken digit recognition task is to separate one specific digit out of other spoken digits. It is common classification task for neural network. The sound samples are processed by network during training and that test samples applied to evaluate performance of constructed neural network. Ideally trained network should capable to recognize sound samples out of dataset used for training. The evaluation parameter is Word Error Rate (WER) which is average number of error made on set of unknown sound samples

The dataset we used in experiments is a subset of the TI46 (The Texas Instruments 46-Word Speaker-Dependent Isolated Word Corpus) speech corpus and consist of ten digits, from zero to nine (or one to ten), each uttered ten times by five different female speakers, resulting in 500 speech fragments sampled at $12.5\,kHz$.

The TI46 speech material was recorded in a low noise sound isolation booth, using an Electro-Voice RE-16 cardoid dynamic microphone, positioned two inches from the speaker's mouth and out of the breath stream.

It is common practice for speech recognition applications to preprocess the sound in order to enhance the speech-specific features to facilitate the recognition process. We utilize *Lyon Passive Ear Model* to transform sound samples into *cochleagrams*.

The Lyon Passive Ear model is a model of the human inner ear or cochlea, which describes the way the human cochlea transforms a sound into a series of neural spikes generated by the hair cells present inside the inner ear. The model consists of a filter bank which closely resembles the selectivity of the human ear to certain frequencies, followed by a series of half-wave rectifiers and adaptive gain controllers both modeling the hair cell response. For example, digit samples from TI46 dataset have sample rate equal to $12.5\,kHz$ and after transformation its cochleagrams have 86 channels.

Pictures in Fig.3 are presenting two examples of cochleagrams from TI46 dataset. It is clear seen that same digit but pronounced by different persons are close to each others. Thus it may be concluded that cochleagrams generated by Lyon ear model enhance speech accented features of spoken material.

3.2 Input signal design

In the experimental setup the input s is $In(t)$. To be more rigorous the input signal $In(t)$ is formed by the information input $I(t)$ modulated by a mask $v(t)$ and multiple by amplitude γ: $In(t) = \gamma v(t) I(t)$. The mask is a periodic function with period τ_D, total delay time. In the case of speech recognition task, the mask is a sparse 2D matrix M with some *sparse density* (typical value is less then 0.1) which determines connections between input sample and neurons of the reservoir: this connection is usually performed through an injection of the input signal into various spatially distributed nodes of a neural network which is forming the reservoir. In the case of a delay dynamics reservoir, this spatial distribution is replaced by a temporal one, spreading the input signal over the time delay, which is viewed as a pseudo spatial

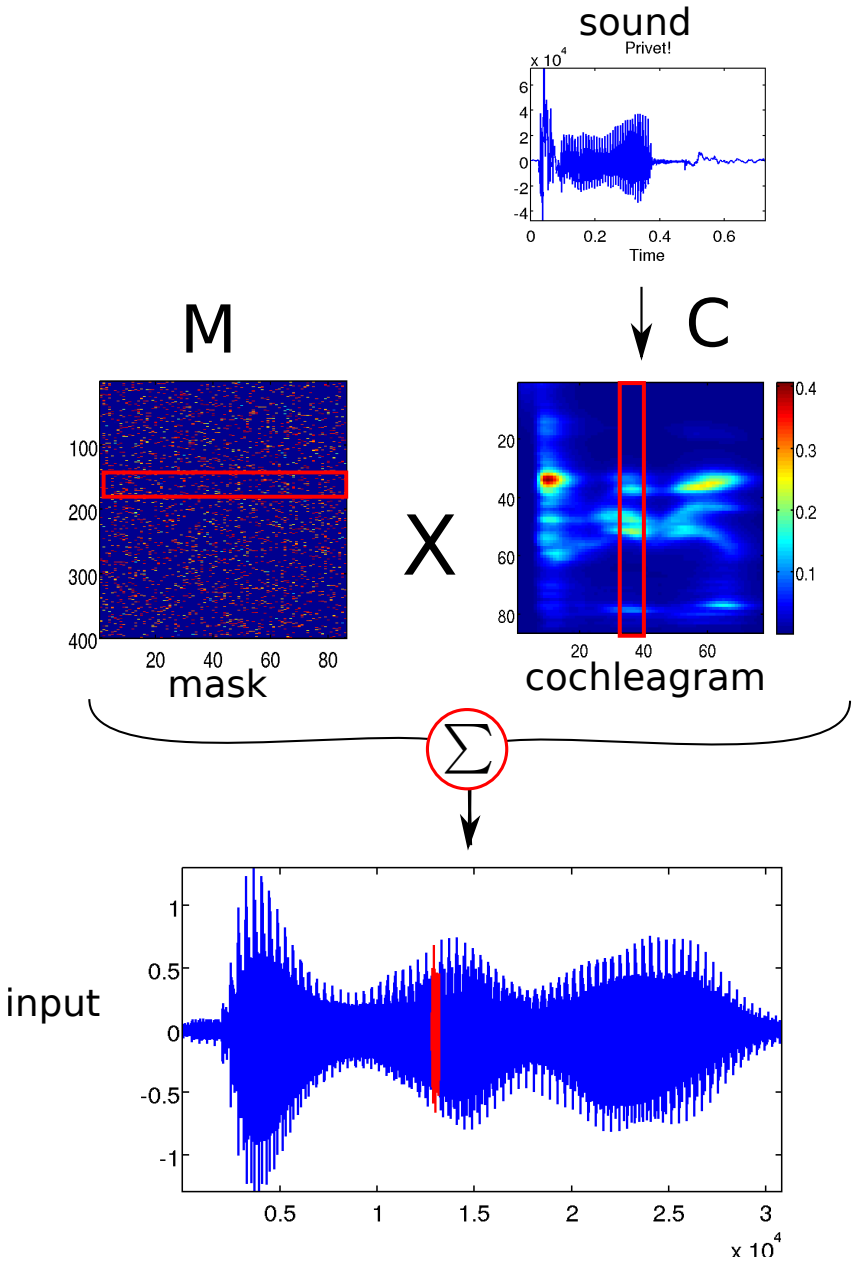

Fig. 4: Input signal design overview.

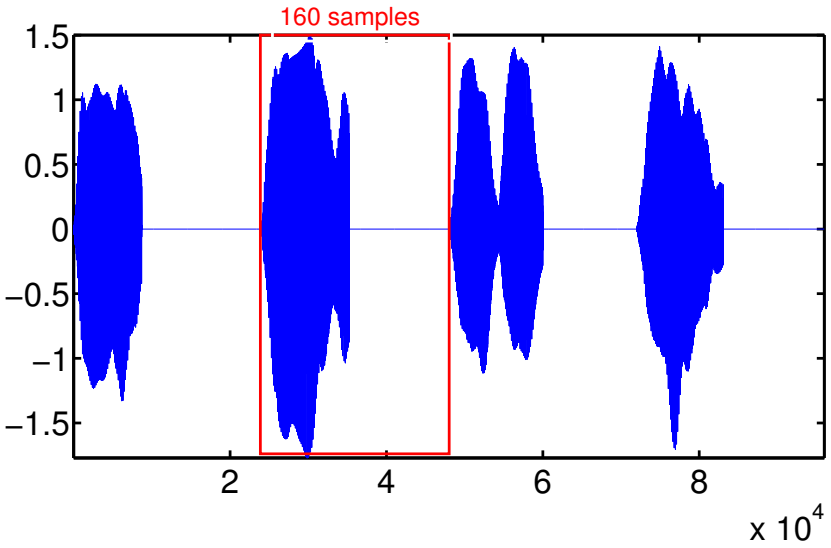

Fig. 5: Example of input data

space. The sparse density of a matrix is the number of nonzero elements divided by the total number of matrix elements. Nonzero elements of matrix is normally distributed with 0 mean and variance 1. The detailed steps to produce the input signal from sound digit is demonstrated below.

Fig.4 demonstrates basic elements of the input signal design including Lyon ear model transformation and masking modulation. The Lyon passive ear model which transforms the input sound signal $d(t)$ into time-frequency 2D matrix $C[n \times 86]$ (where n is width of a digit in terms of 2D cochleagram, typical values for n are in range from 50 to 100) is more or less clear. After this transformation the input signal is represented by a cochleagram C with 86 frequency channels (spanning along the vertical axis in 4), and with a number of samples in time which depends on the spoken word (in the range of $50-100$ along the horizontal axis). The mask $M[86 \times N_{nodes}]$ is a sparse matrix (with many 0's) of randomly distributed values -1 and 1. To produce N_{nodes} samples (one for single node) from columns of the cochleagram C one should multiply it by the matrix M,

$$In = M \times C.$$

To produces 1D signal from elements of In one should transform it in one-dimensional vector by concatenating its columns (see Fig.4). Total number of elements of input signal for one specific digit is equal to $N_{nodes} \times n$, where n is the number of temporal samples after Lyon model transformation. To make the situation more clear let's make simple calculation on number of input samples needed for one spoken digit to be inserted in the reservoir. Typical size of samples after Lyon model transformation ranges from 50 to 100. Let for example $n = 70$ and number of nodes $N_{nodes} = 400$. Number of samples is equal to $400 \times 70 = 28000$.

To process all 500 spoken digits in experiments it is needed to combine several digits in one input signal. For convenient post-possessing all digits can be extended by zero value up to length of 160 samples each. It is done due to the longest digit has length 130. Hence to precess all digit automatically one need gap between digits. It presented case the smallest gap is 30 samples. Space between digits also serve as a time when reservoir system goes to a rest state. For above example it implies that each digit has $400 \times 160 = 64000$ points in resulting input signal and total length of the input data file contains $64000 \times N_{samples}$, where $N_{samples}$ is number of samples for one input data file. Typical input data file is shown in Fig.5 with bunch of 4 spoken digits.

3.2.1 LSM reservoir states

Liquid State Machine analysis of experimentally obtained results were made in framework of the MATLAB `Reservoir Toolbox` designed by Ghent University `http://reslab.elis.ugent.be/rctoolbox`. It means that we should generate reservoir states for each time step from recorded experimentally time traces. The state of reservoir in time t is the state of all nodes of the system. The state of reservoir can be described by the expression,

$$\{x(t - k \cdot \delta\tau_D)\}_{k=1\ldots N_{nod}} \tag{2}$$

In other words, the state of reservoir is the set of time-traces for all nodes. In simulation it is a matrix with many columns. It is obvious that reservoir states renew all its elements after total time delay τ_D.

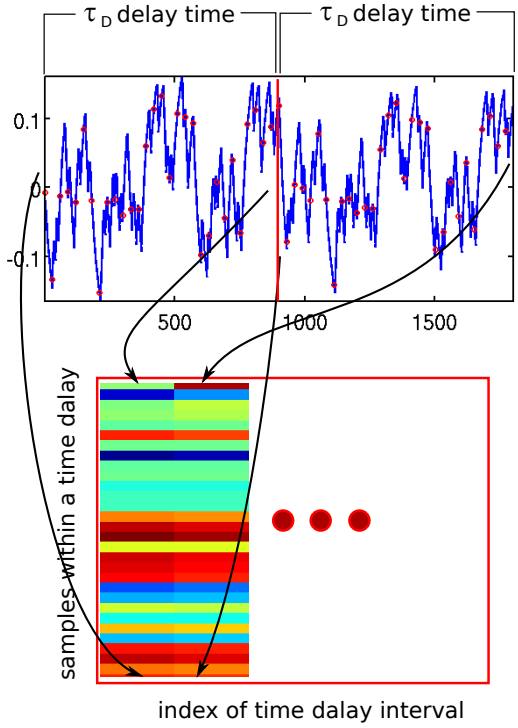

Fig. 6: Reservoir states collection procedure

It means that to collect all information about reservoir dynamics it is enough to record reservoir states separated by τ_D. Fig. 6 shows the process of formation of a reservoir state.

The time-trace recorded from an experiment is not ready to process directly. It has several spoken digits at once. In addition it is oversampled. Typical time trace is shown in Fig.7 Hence all digits have same length (160 cochleagram samples extended by same mask) it is easy to cut a time-trace in equal parts to have separate traces for every single digit. The oversampling rate can be calculated form the obtained single part. If time-trace from for single spoken digit has N_{digit} points then oversampling rate is the ration between N_{digit} and $160 \times N_{neur}$, $\hat{R}_{over} = N_{digit}/(160 \times N_{neur})$. Practically the oversampling rate is not integer number. Let us denote R_{over} an upper integer approximation of $\hat{R}_{over}$. Down-sampling can be done by simple linear interpolation procedure. MATLAB embedded function can be utilized. However we use a little different approach. Time-trace for single digit is interpolated to have exact $160 \times N_{neur} \times R_{over}$ number of points. It means that each node of reservoir is represented by R_{over} points. Down-sampling can be done by selecting specific point among R_{over} points or through more advanced averaging procedure. We use specific point selection method. Last step in processing is to cut off zero part and fold trace into reservoir state matrix by above described procedure (see Fig.6).

The code example for processing raw time-trace is presented below.

```
digtlen = load('../../inputs/load_lengthfile1.dat');

oversampl = 21; % oversampling rate previously calculated
shft = 1;
neur = 150;
numofsample = 16; % number of digits in one file
start  = 118000;  % start point of first digit
cutlen = 404018;  % length of one digit, priviously calculated

numoffiles = ceil(500/numofsample); % number of files to cover all 500 digits



for num = 1:numoffiles,
    len = numofsample;
    if num == numoffiles, % last file cantain less then  'numofsample' digits
        len = 500-numofsample*(numoffiles-1);
    end
    % load raw timetrace from files in current folder
    [resp0,t]= wfm2read(['response' num2str(num,'%3.3u') '.wfm']);
    % initial cut
```

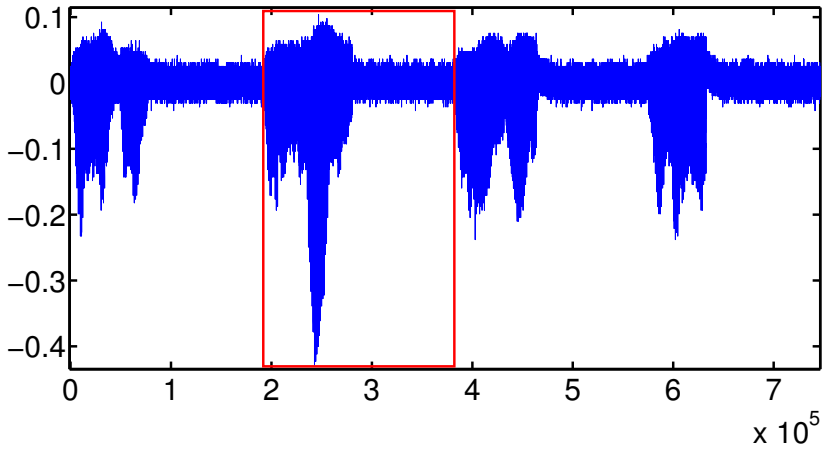

Fig. 7: Raw time-trace recorded form experiment.

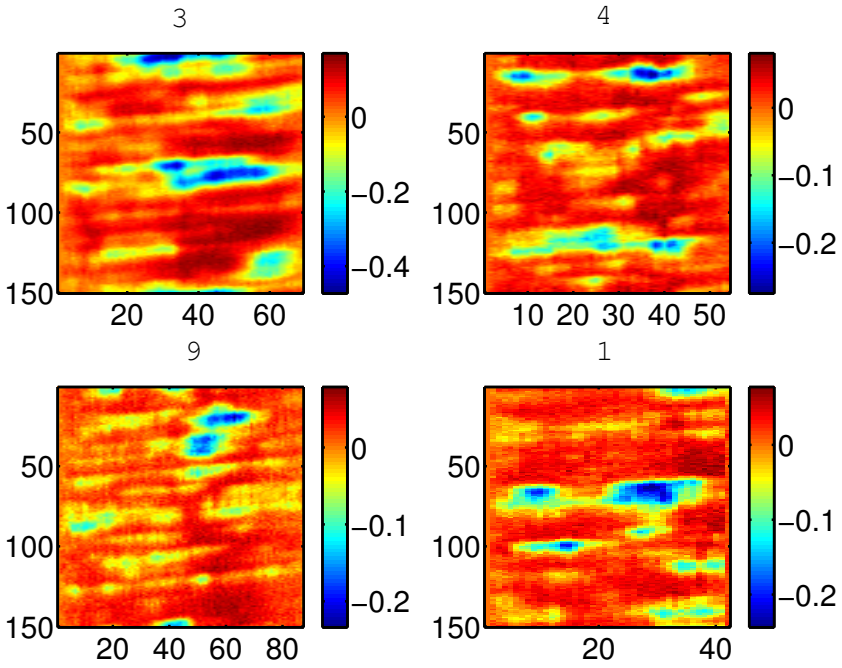

Fig. 8: Reservoir states examples

```
22      resp = resp0(start:start+cutlen*numofsample);
23      expdatalength = length(resp);
24      t_exp = linspace(1,expdatalength,len*oversampl*neur*130);
25      % interpolate
26      resp = interp1(1:expdatalength,resp,t_exp);
27
28      % represent each digit as column of matrix
29      respMat = reshape(resp,[],len);
30
31      for i =1:len,
32          dignum = (num-1)*numofsample+i;
33           % cut to single digit
34          save(['reservoir/output' num2str(dignum) '.dat'],'digt','-ASCII');
35          digt =respMat(1:oversampl*neur*digtlen(dignum),i);
36
37      end
38  end
```

Examples of reservoir states extracted from time-traces are shown in Fig.8

The procedure of read-out is outlined in Fig.9. There are three main steps: 1) applying the W_{opt} and getting raw read-out; 2) applying first post-processing step which in fact integrate the result over time

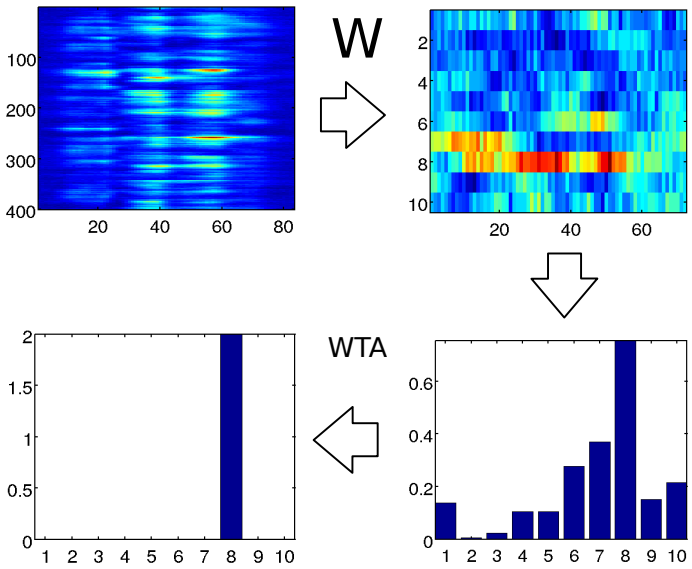

Fig. 9: Read-out

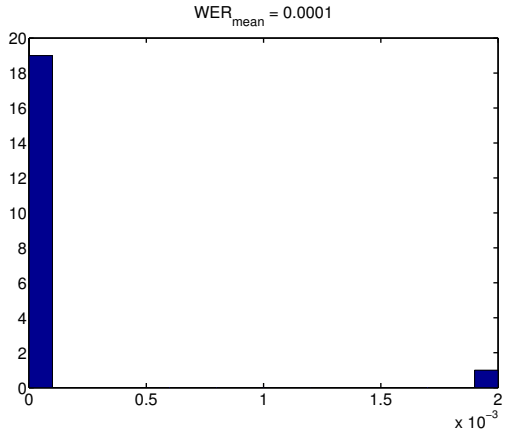

Fig. 10: The statistics on cross-validation partition

axis (horizontal line); 3) applying function $winner$-$take$-all-non-linearity, which selects best value from previously obtained elements. Final result of read-out is binary array with 0's everywhere except one element with1 which indicate the answer.

The scoring is simple. After read-out every sample from test subset produce an answer. To collect all errors one should compare answers and targets. If they are equal, the error is 0. If not the error is 1. Total score collects all errors in test subset,

$$score = \frac{(0+1+\cdots+1+0)}{number\ of\ samples}$$

Finally Word Error Rate (WER) is just mean value of scores taken over all folds (cross-validate cycles),

$$WER = \mathrm{mean}(score)$$

3.3 Experiments

We performed first test and obtained best $WER = 0.0001(0.01\%)$. The statistics on cross-validation partitions are shown in Fig.10 It is obvious that value 0 dominates. It means that the constructed system gives errors in recognition of speech sample (spoken digit) in less then one percent cases. Our experiments are mostly preliminary. More detailed investigation needed to find best operating conditions. Nevertheless even now it gives very promising result. For comparison, we made direct training (numerically) the constructed reservoir on input signal (roughly speaking cochleagrams after masking treated as reservoir states skipping nonlinear system). The best achievement on direct training gives $WER = 0.06$ (6%) which is about two order worse then with the delay system.

4 Conclusion

In this paper we demonstrate that reservoir dynamics can be realized by elector-optical setups with dynamical delay. Dynamical delay systems have the inherent property that their phase space is infinite-dimensional. Therefore, we can achieve high-dimensionality with a small number of electro-optical components and explore how to benefit best from this peculiar property.

In this work the electro-optical coupled system with delays performed by programmable device (FPGA) is investigated. It was shown that the electro-optical setup are well developed hardware for delay system implementation with good reliability and controllability and can operate as reservoir computer. It was proven by simulation and providing experiments that electro-optical systems give very good result in main 'brain computing' benchmark: Spoken Digit Recognition task. The Spoken Digit Recognition task is very important cognitive task for neural network. Our electro-optical setup can perform recognition task with the word error rate (WER) less then 0.01%. It should be mentioned that up to now there are many system with capability to recognize speech but most of them use high computation power. In contrast our model need only small portion of simulation work and most part of the system works as analog device based on physical principles. It make the system be close to brain in sense of analog information processing.

References

[1] W. Maass, T. Natschläger, and H. Markram. Neural Comp. 14, 2531, 2002.

[2] T. Natschläger, W. Maass, and H. Markram. Special Issue on Foundations of Information Processing of TELEMATIK 8, 39, 2002.

[3] H. Jaeger. Technical Report GMD Report 148, German National Research Center for Information Technology, 2001.

[4] W. Maass and H. Markram, J. Comp. & Sys. Sci. 69, 593, 2004.

[5] Lyon, R. (1982). A computational model of filtering, detection and compression in the cochlea. In Proceedings of the IEEE ICASSP, pages 1282–1285.

[6] 'Auditory Toolbox' can be found here `http://cobweb.ecn.purdue.edu/~malcolm/interval/1998-010/`.

[7] MATLAB `Reservoir Toolbox` designed by Ghent University `http://reslab.elis.ugent.be/rctoolbox`

[8] Verstraeten, David (2009). Reservoir Computing: Computation with Dynamical Systems, Electronics and Information Systems, Gent, Ghent University, pp. 190 `http://organic.elis.ugent.be/sites/www.reservoir-computing.org/files/sites/organic.elis.ugent.be/files/dvrstrae_organic.pdf`

[9] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. J. Royal. Statist. Soc B., 58(1):267–288.

[10] Dutoit, X., Schrauwen, B., Van Campenhout, J., Stroobandt, D., Van Brussel, H., and Nuttin, M. (2009). Pruning and regularization in reservoir computing. Neurocomputing, 72:1534–1546.

[11] Tikhonov, A. and Arsenin, V. Y. (1977). Solutions of Ill-Posed Problems. Winston and Sons.

[12] F.T. Arecchi, G. Giacomelli, A. Lapucci, R. Meucci}, Two-dimensional representation of a delayed dynamical system, \emph{Phys. Rev. A} \textbf{45}, R4225 (1993).

[13] J.-P. Goedgebuer, L. Larger, H. Porte}, Optical cryptosystem based on synchronization of hyperchaos generated by a delayed feedback tunable laserdiode, \emph{Phys. Rev. Lett.}, \textbf{80}, 2249–2253 (1998).

[14] Moody, J. (1992). The effective number of parameters: An analysis of generalization and regularization in nonlinear learning systems. In Proceedings of NIPS, volume 4, pages 847–854.

[15] wyffels, F., Schrauwen, B., and Stroobandt, D. (2008a). Stable output feedback in reservoir computing using ridge regression. In International Conference on Artificial Neural Networks.