2009 Ninth International Conference on Intelligent Systems Design and Applications

Echo State Network for Abrupt Change Detections in Non-stationary Signals

Qingsong Song[1,2], Zuren Feng[1,2], Liangjun Ke[1,2], Min Li[3]
[1]System Engineering Institute, Xi'an Jiaotong University. [2]the State Key Laboratory of Manufacturing Systems Engineering, Xi'an Jiaotong University, Xi'an 710049, China. [3]College of Mathematics and Information Science, Shaanxi Normal University, Xi'an 710062, China.
qssong@sei.xjtu.edu.cn; fzr9910@xjtu.edu.cn; kelj@163.com

Abstract

The issue of abrupt change detection (ACD) in non-stationary time series signal is considered as a signal classification problem in this paper. A novel reservoir-computing based neural network model (RCNN) is proposed. The main component of RCNN is a large size, sparsely and randomly interconnected dynamical reservoir (DR), which is followed by a single layer perceptron (SLP). The signal containing abrupt changes is firstly projected into the high dimensional state space of DR, and then is linearly classified by the SLP. The SLP is trained by the delta leaning rule. The classification brought out by the SLP is the ACD result. Two synthetic non-stationary time series signals, one is non-chaotic, another one is chaotic, are verified on the RCNN respectively. The simulation experiment results show that the ACD performance of the proposed RCNN is comparable with that of the segment function embedded in MATLAB for the non-chaotic signal, and even outperforms for another chaotic signal. It is concluded that RCNN is a more efficient ACD technique.

1. Introduction

Many signals are non-stationary due to some noises or disturbances, that is, the signals are composed of segments of stationary behavior with abrupt changes in their characteristics in the transitions between different segments. Detection of abrupt changes in signals' characters has a significant role in many applications, such as in failure detection and diagnosis systems [1], in condition-based maintenance of industrial processes [2] and in abrupt change detection-based signal segmentation problems etc [3,4].

There have been many different approaches to deal with abrupt change detection (ACD) problems. If those disturbances can be modeled as Gaussian random process with a constant variance or some continuous time-varying variance, the corresponding ACD problem can be well done by Kalman filter or some special case of the Kalman filter [5,6]. However, if the disturbance no longer is Gaussian, the ACD problem would be faced with much more difficulty to deal with. Some methods originated from nonlinear system identification are adopted to solve this later kind of ACD problem, such as adaptive forgetting through multiple models [7], the mixture of experts or modular neural networks combined with a gating network [8,9], and some CUSUM-type algorithms [10].

Aiming at ACD problem, a novel method, a reservoir-computing based neural network model (RCNN) is proposed in this paper. A typical implementation of reservoir-computing method [11] is echo state network (ESN) [12]. ESN has a significant improved performance compared with other common methods when it was used to predict some chaotic time series, to control automobile robot movement [12,13]. To our knowledge, however, there have not been any reports on how reservoir-based neural computing would do with ACD problems.

This paper is organized as follows. In Section 2, the proposed RCNN for ACD problem and its corresponding adaptation algorithm are described in details. In Section 3, two synthetic signals with abrupt changes are processed by the proposed method, the effectiveness of the method to ADC are verified. Discussions and conclusions are made in Section 4 and Section 5 respectively.

2. Methods

2.1 Dynamical Reservoir

The architecture of the proposed RCNN is mainly consisted of three parts. Among these three parts, two parts are described with more details in this paper. One is the large scale dynamical reservoir (DR), which projects the external stimulations into a high dimensional state space. Another part is the readout, which extracts the useful information from the DR and then linear or nonlinear combines them into the desired output.

As stated in reference [12], The DR, which is the core component of ESN, is constituted of many individual neurons with nonlinear dynamics. These neurons are interconnected randomly and sparsely. The connectivity density of DR usually falls in the range from 0.04 to 0.2. The spectral radius is chosen to be less than unit so as to guarantee the echo state property [14].

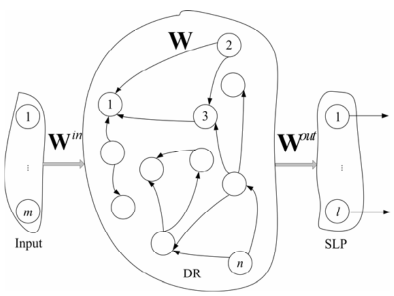

Fig.1. The architecture of the proposed reservoir-computing based neural network model.

During training process, it is only necessary to adjust connection weight between readout neuron and each individual within DR. DR does not need adaptation any more after being initialized. Due to the sparse connectivity, the dynamical process of each unit within DR can display certain systematic variant of input signal. So DR has powerful representation capacity when it is stimulated by external stimulus, simple linear regression method is competent for the training task.

The DR of RCNN is a discrete-time DR which has m input neurons and is made up of n sigmoid neurons. The connection weights between each input neuron and each neuron within DR are given in a $n \times m$ matrix $\mathbf{W}^{in}$. The connection weights of DR are given in a $n \times n$ matrix $\mathbf{W}$. The state of DR $\mathbf{X}$ is computed as (1).

$$\mathbf{X}(k+1) = \tanh(\mathbf{W}^{in} \cdot u(k+1) + \mathbf{W} \cdot \mathbf{X}(k)) + \mathbf{v}(k+1) \qquad (1)$$

where u is external input stimulus, $\mathbf{v}$ is some white noise, $\tanh$ is hyperbolic tangent activation function of neuron with DR, k is time step of the simulation experiment.

2.2 Single Layer Perceptron

The readout of the proposed RCNN is a single layer perceptron (SLP) [15]. In this paper, the number of the types of abrupt changes is known in advance, and is noted as a number l. For a specific task, the number of neurons which make up of the SLP is equal to the number of the types of abrupt changes contained in the non-stationary signal.

$$\mathbf{Y}(k+1) = \mathrm{hardlim}(\mathbf{W}^{out} \cdot \mathbf{X}(k+1) + \mathbf{b}) \qquad (2)$$

SLP receive the outputs of all units within DR. The corresponding connection matrix is noted as $\mathbf{W}^{out}$, which has a size of $l \times n$. The trained LSP output $\mathbf{Y}$, which is a $l \times 1$ dimensional vector, is calculated as (2), where $\mathrm{hardlim}$ is hard limit transfer function of SLP neuron, $\mathbf{b}$ is bias vector with size $l \times 1$.

Fig. 1 illustrates the architecture of RCNN. The three parts consisting of the RCNN are input, DR and SLP, which is respectively labeled Input, DR, and SLP as shown in Fig. 1.

2.3 the Abrupt Change Detection Method

The ACD algorithm of RCNN is as following steps.

1) Generate Train Set and Test Set: For a given system with switching dynamic modes, an output sequence set can be obtained by assigning the system with different initialized values. Then all obtained output sequences in this set are scaled and shifted individually. The set is divided into two sets, one set for train, and the other one set for test.

2) Generate Network: Based on the given system, the values of m, n and l are assigned at first. The connectivity density ρ and spectral radius R of DR are also assigned initially. In fact, these two parameters, ρ and R, are crucially important for the eventual success of RCNN training. In order to choose right values of ρ and R, it might need tries many times in practice. The input weight matrix $\mathbf{W}^{in}$ is random generated. The function *sprand* embedded in MATLAB is used to generate the DR weight matrix $\mathbf{W}$, the third parameter of function *sprand* is set to the value of ρ. Then, the resulted $\mathbf{W}$ is adjusted to has a spectral radius with the value of R.

3) Collect DR State: Each sequence within the train set is forced into the DR step by step. According to (1), DR state is calculated. After an initial washout time T_0 (T_0 is usually chosen to be larger than the value of n), every DR state is collected in a matrix $\mathbf{M}$. If the length of train set is T, the matrix $\mathbf{M}$ has a size $(T - T_0 + 1) \times n$.

4) Train SLP: Since the time instants when abrupt change happens are known in advance in the train set, the desired SLP outputs $\mathbf{Y}_d$ can be obtained. $\mathbf{Y}_d$ has a size $(T - T_0 + 1) \times l$. The pairs $(\mathbf{X}(k), \mathbf{Y}_d(k))$ which are to be used to train the SLP are collected, $k = (T_0+1), (T_0+2), \cdots, T$. The widely used δ learning rule is employed to adapt the SLP. The output weight matrix $\mathbf{W}^{out}$ is then computed.

It is well known that if the δ learning rule can complete the adaptation process of matrix $\mathbf{W}^{out}$ in a finite number of updating time step, the inputs to SLP, i.e. $\{\mathbf{X}(k)\}$, $k = (T_0+1), (T_0+2), \cdots, T$, must have the linear separability in the high dimensional state space of DR. The experiments in later Section show that the δ learning rule implemented here can indeed complete the train process in a finite number of updating time steps.

5) Test Network and Evaluate its Detection Performance: The individual within the test set runs on the trained RCNN, which is the DR combined with the trained SLP. Since the time instants when abrupt changes occur in test set are assumed to be known in advance, the evaluation on detection performance of the trained RCNN can be made by comparing the desired output of $\mathbf{Y}_d$ and the real output of $\mathbf{Y}$.

Four intuitive performance indexes to evaluate ACD algorithms are usually used. They are rate of right detection, rate of false detection, rate of missed-detection, and accuracy of detected change time. The detailed definitions are described in APPENDIX.

3. Experiments and results

Two non-stationary signals are artificially generated. The generated signals are scaled and shifted to fall into a range [-0.4, +0.4] in order to keep the activation of sigmoid neuron in DR away from its saturation zone, and then to be verified on our proposed RCNN method.

For both tasks the entities of input weight matrix are uniform random numbers ranging from -0.5 to +0.5. The DR has a size 50 (n=50). Its connectivity density is 0.2 (ρ=0.2) and its spectral radius is 0.4 (R=0.4). The state noise of DR is a white noise with a mean 1e-5. There is no feedback connection from SLP to DR.

3.1 Task A: Synthetic Random Signal with Abrupt Spectral Changes

Three two order IIR digital filters are designed. The system function $\mathrm{H}(z)$ of an IIR filter is (3), where b_1 and b_2 are free coefficients, G stands for gain [16].

$$\mathrm{H}(z) = \mathrm{G}/(1 + b_1 z^{-1} + b_2 z^{-2}) \tag{3}$$

Each one has a different 3dB cutoff frequency (f_c) with each other. Those f_c s are 0.2, 0.5 and 0.8. It means that l should be equal to three, i.e., the SLP has three outputs, one represents the filter with f_c equal to 0.2, one represents the filter with f_c equal to 0.5, and one represents the filter with f_c equal to 0.8.

All filters are fed with the same one white noise time series. The output of one filter is switched to the output of the second filter at a defined time instant t_1. The output of the second filter is switched to the output of the third filter at another defined time instant t_2. The total length of the signal is 700 samples of which the first 100 samples are used to remove the effect that arbitrary initial DR state might make, i.e. T=700, T_0=100. In the left 600 samples, the first 200 samples are from the first filter, the following 200 samples from the second filter, and the last 200 samples from the third filter, i.e. t_1=200, t_2=400. In order to train and test RCNN, a total 150 signals are generated according with the same way as above. Among them, 50 signals are chosen to train the model; the left are chosen to test the trained model.

It means that there are 100 times of independent test experiment being done in the test period. As a comparison, the same test set is verified by the function *segment* [17] embedded in MATLAB. Fig.2 shows the abrupt changes detection result of a test run which precisely detects the time instants when the abrupt changes occur. Table 1 shows the statistical performance indexes which are evaluated on the results of all 100 independent test experiments. It can be seen from Fig.2 and Table 1 that the proposed method satisfactorily fulfills this ACD task. In addition, the performance that the proposed RCNN method corresponds with is comparable with the performance the function *segment* corresponds with.

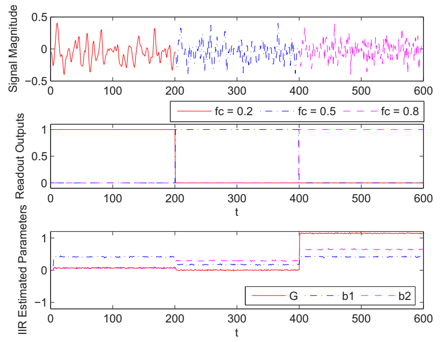

Fig. 2. The abrupt changes detection result of a test run which precisely detects the time instants when the abrupt changes occur. Three sub-graphs are included. The upper sub-graph shows an instance of the signal in test set. The results obtained by the proposed method and the *segment* function in MATLAB are shown in the middle sub-graph and in the lower sub-graph separately.

TABLE 1
STATISTICAL PERFORMANCE INDEXES EVALUATED ON THE ABRUP CHANGE DETECTION RESULTS OF ALL THE INDEPENDENT TEST RUNS

	Rate of Right Detection	Rate of False Detection	Rate of Missed-Detection	Accuracy of Detected Change Time
RCNN	98%	1%	1%	200; 399
Segment	96%	4%	0%	201; 400

3.2 Task B: the Mackey-Glass Time Series with Abrupt Time-delay Changes

The Mackey-Glass (MG) time series is a typical chaotic time series, and is often used to be a benchmark time series to be predicted [18]. The differential equation of the MG chaotic time series is (4).

$$dz/dt = 0.2z(t-\tau)/(1+z(t-\tau)^{10}) - 0.1z(t) \quad (4)$$

where z represents state and τ represents a time delay. If τ is equal to or above 16.8, chaos will occur. The function *dde23* with the default settings in MATLAB is chosen to solve above delay differential equation.

The value of parameter, time-delay τ, is chosen to be 17, 23 and 30 successively. The corresponding equations with different initial value are solved simultaneously. The output of the equation with τ =17 is switched to the output of the equation with τ =23 at a defined time instant t_1, and then switched to the output of the equation with τ =30 at another defined time instant t_2. Since there are three distinct values of τ, the SLP has three outputs, i.e. l equals three. Each output corresponds to a MG dynamic system with a value of τ.

For task *B*, the total length of the signal is 1,000 samples i.e. T =1,000. The first 400 samples correspond to the equation equipped with τ =17, of which the first 100 samples run freely to eliminate the effect that the initialized arbitrary DR state might bring out, i.e. T_0 =100. The following 300 samples correspond to the equation equipped with τ =23, and the last 300 samples correspond to the equation equipped with τ =30. So the defined time instants when abrupt change occurs are that t_1 =300, t_2 =600.

In order to train and test the proposed method in this paper, a total number of 150 signals are generated with different initial values, and then split into two sets: 50 signals are chosen to train the model, they consist of the train set; another 100 signals are chosen to test the trained model, they consist of the test set.

As in task *A*, the same test set is verified by the function *segment* embedded in MATLAB for comparison. The results are illustrated with Fig.3 and Table 2. It is demonstrated that the proposed method has superiority in all of these four performance indexes over the function *segment* for this task.

The function *segment* embedded in MATLAB is recruited to segment data and to track abruptly changing systems. *Segment* regards the segment problems or the ACD problems as a kind of nonlinear system identification problem. The jumping parameters or parameters that change in an irregular fashion are estimated by multiple Recursive Least Squares (RLS) algorithms running in parallel, where each RLS has a corresponding weighting factor [7,17].

Compared with *segment*, the proposed method has a salient feature, i.e., the RCNN does not estimate any specific parameters about the dynamic "source" which generates the non-stationary signals, such as G, b_1, b_2 in task *A*, τ in task *B*, but directly detects the time instant when the abrupt change occurs such as t_1, t_2. In other words, the proposed RCNN is an ACD method which entirely depends on the input time series signal, no information on the model which might describes the input signal is required.

Moreover, the proposed method can also recognize the context behind the abrupt change event. For example, the abrupt change event occurring on time instant t_1 =200 in task *A* implies a context that the "source" described by the filter with f_c =0.2 is switching to the "source" described by the filter with f_c =0.5, the abrupt change event occurring on time instant t_2 =600 in task *B* implies a context that the "source" described by the MG dynamic system with τ =23 is switching to the "source" described by the MG dynamic system with τ =30.

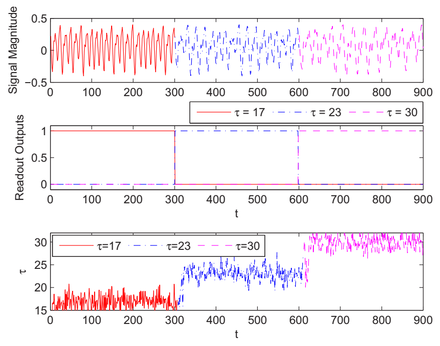

Fig. 3. The abrupt changes detection result of a test run which precisely detects the time instants when the abrupt changes occur. Three sub-graphs are included. The upper sub-graph shows an instance of the signal in test set. The results obtained by the proposed method and the *segment* function in MATLAB are shown in the middle sub-graph and in the lower sub-graph separately. The legend 'τ =17' indicates a MG dynamic system with τ =17, and so on.

TABLE 2
STATISTICAL PERFORMANCE INDEXES EVALUATED ON THE ABRUPT CHANGE DETECTION RESULTS OF ALL THE INDEPENDENT TEST RUNS

	Rate of Right Detection	Rate of False Detection	Rate of Missed-Detection	Accuracy of Detected Change Time
RCNN	93%	5%	2%	300; 598
Segment	83%	14%	3%	315; 618

4. Discussions

4.1 Diversity of the Response of DR to External Stimulus

The response trace of four random selected units within DR during a train run for task *A* and for task *B* is illustrated with Fig. 4 and Fig. 5 respectively. The dynamical process of the units within DR displays certain systematic variant of the corresponding input signals. They have diversified dynamics and are slightly correlated with each other [19]. It is just the sparse DR with the rich states induced by external stimulus, which is called the "reservoir of rich dynamics" in reference [20], that contain sufficient information for the SLP to conveniently extract to deal with the ACD problems. In other word, the sparsely interconnected and large size DR boosts the classification ability of SLP indeed.

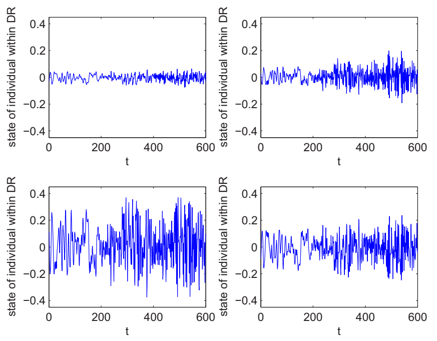

Fig. 4. The response trace of four random selected individuals in DR during a train run for task *A*.

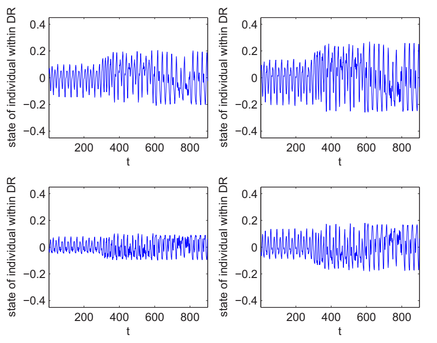

Fig.5. The response trace of four random selected individuals in DR during a train run for task *B*.

4.2 The Stability Issue

The proposed RCNN has not any feedback from the SLP output to the DR or to the input part. It is a purely input-driven dynamical system indeed. So the stability of the proposed method can be ensured due to the echo state property [14]. In fact, during the experiment process for both tasks, the divergent phenomenon of the dynamical behavior of units with DR is not observed all along.

4.3 The Convergence Issue

It is known that if the δ learning rule for a perceptron can approach its convergence in a finite number of updates, it is required that the data being processed must be linearly classifiable. Separation property, which addresses the amount of separation between the trajectories of DR states caused by two different input streams, has been proofed to be one of the necessary and sufficient conditions for powerful computing with liquid state machine (LSM) [21]. LSM is another successful implementation carrying out the reservoir-computing method. As for the proposed RCNN, the phenomenon, that the δ learning rule for SLP can complete its adaptation in no more than sixty updates in the train runs with the both tasks, has been illustrated with computer simulation experiments, however, the theoretical analysis, that the external stimulus is linearly classifiable in the high dimensional DR state space, has not been made out.

4.4 The Issue of Generalization Ability

Since the ACD problems are essentially treated as a kind of signal classification problem in this paper, the margin [22] corresponding to the trained classifiers (i.e. SLP) is crucial to the generalization ability of the proposed RCNN. A successful trained SLR without the maximum margin is not the best one. It might bring out false detection results during test. The false detections and missed-detections list with Table 1 and Table 2 might blame on the little margin corresponding to the trained SLP. In order to perfect the RCNN method, the SLP of RCNN is to be substituted by a maximum margin linear classifier such as SVM in our prospective works.

5. Conclusions

As far as we are concerned, the diversity of the dynamical response of the units within the sparsely interconnected DRs driven by external stimulus is successfully exploited to promote the classification ability of the following SLPs. The proposed RCNN method can be said to be an effect technique to deal with some ACD problems as demonstrated with these simulation experiments.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant No. 60875043 and the State Key Development Program for Basic Research of China under Grant No. 2007CB311006.

Appendix: the Detailed Definitions on the Performance Indexes

The symbols are involved in the definitions list as follow:

t : an instant when abrupt change occurs in test signal.

$\hat{t}$: the estimated instant corresponding with t.

n_t : the total number of independent test runs.

ε_t : a pre-defined tolerance deviation of $\hat{t}$.

$\#\hat{t}$: the number of independent test runs which have the same $\hat{t}$.

Rate of Right Detection (r_r)

For a test run, if $\hat{t} \in [t-\varepsilon_t, t+\varepsilon_t]$, this test run is said a run with right detection. After all test runs are completed, r_r is statistically computed as $r_r = (\#\hat{t})/n_t$.

Rate of False Detection (r_f)

For a test run, if $\hat{t} < (t-\varepsilon_t)$ or $(t+\varepsilon_t) < \hat{t} < \infty$, this test run is said a run with false detection. After all test runs are completed, r_f is statistically computed as $r_f = (\#\hat{t})/n_t$.

Rate of Missed-Detection (r_m)

For a test run, if $\hat{t} \to \infty$, this test run is said a run with missed detection. After all test runs are completed, r_m is statistically computed as $r_m = (\#\hat{t})/n_t$.

The sum of r_r , r_f and r_m is equal to unit.

Accuracy of Detected Change Time (t_a)

A mapping (f_t) is defined, $f_t : \hat{t} \to \#\hat{t}$. There exists a value (t_a) which satisfies that $f_t(t_a) = \max(f_t(\hat{t}))$. It is the t_a which indicates the accuracy of detected change time.

For the two tasks, ε_t is chosen to be a value of 30.

References

[1] J. J. Gertler, "Survey of model-based failure detection and isolation in complex plants," IEEE Control Systems Magazine, vol. 8, issue. 6, pp. 3-11, Dec. 1988.

[2] M. Paavola, M. Ruusunen, and M. Pirttimaa, "Some change detection and time-series forecasting algorithms for an electronics manufacturing process," Control Engineering Lab., University of Oulu, Report A No 26, March 2005.

[3] A. Ilin, H. Valpola, and E. Oja, "Nonlinear dynamical factor analysis for state change detection," IEEE Transactions on Neural Networks, vol. 15, no. 3, pp. 559-575, May 2004.

[4] A. Ukil, R. Zivanovic, "Automatic signal segmentation based on abrupt change detection for power systems applications," In *2006 IEEE Power India Conference*.

[5] E. Cortina, D. Otero, and C. E. D'Attellis, "Maneuvering target tracking using extended Kalman filter," IEEE Transactions on Aerospace and Electronic Systems, vol. 27, issue. 1, pp. 155-158, Jan. 1991.

[6] X. R. Li, Y. Bar-Shalom, "Design of an interacting multiple model algorithm for air traffic control tracking," vol. 1, issue. 3, pp. 186-194, Sep. 1993.

[7] P. Andersson, "Adaptive forgetting in recursive identification through multiple models," Int. J. Control, vol. 42, no. 5, pp. 1175–1193, 1985.

[8] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," Neural Computation, vol. 3, issue. 1, pp. 79-87, 1991.

[9] A. Kehagias, V. Petridis, "Predictive modular neural networks for unsupervised segmentation of switching time series: the data allocation problem," IEEE Transactions on Neural Networks, vol. 13, no. 6, pp. 1432-1449, Dec. 2002.

[10] M. Basseville, I. Nikiforov, *Detection of Abrupt Changes – Theory and Applications*. Ser. Informaton and system science. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[11] D. Verstraeten, B. Schrauwen, M. D'Haene, and D. Stroobandt, "An experimental unification of reservoir computing methods," Neural Networks, vol.20, issue. 3, pp. 391-403, April 2007.

[12] H. Jaeger, H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless telecommunication," Science, vol. 304, no. 5667, pp.78-80, April 2004.

[13] E.A. Antonelo, B. Schrauwen and D. Stroobandt, "Event detection and localization for small mobile robots using reservoir computing," Neural Networks, vol. 21, issue. 6, pp. 862-871, Aug. 2008.

[14] H. Jaeger, "A tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the "echo state network" approach," International University Bremen, GMD Report 159, 2002.

[15] S. Haykin, *Neural networks – A Comprehensive Foundation*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1999.

[16] S. Haykin, *Adaptive Filter Theory*, 4th ed. Englewood Cliffs, NJ: Prentice-Hall, 2001.

[17] L. Ljung, *Matlab System Identification Toolbox 7 User's Guide*, The MathWorks, Inc., 2008.

[18] M. C. Mackey, L. Glass, "Oscillation and chaos in physiological control systems," Science, vol. 197, no. 4300, pp. 287-289, July 1977.

[19] M. C. Ozturk, D. Xu, and J. C. Principe, "Analysis and design of echo state. networks," Neural Computation, vol. 19, no. 1, pp. 111-138, 2007.

[20] H. Jaeger, "The echo state approach to analyzing and training recurrent neural networks," German National Research Center for Information Technology, Tech. Rep. No. 148, 2001.

[21] W. Maass, T. Natschläger, H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," Neural Computation, vol. 14, no. 11, pp. 2531-2560, Nov. 2002.

[22] T. Jaakkola, Computer Science and Artificial Intelligence Laboratory, MIT, Lecture 2 of the courses Machine Learning (Fall 2008) [online]. Available: http://courses.csail.mit.edu/6.867/lectures/l2.1.notes.pdf.