

Temporal Sequence Processing using Recurrent SOM

Timo Koskela
Markus Varsta
Jukka Heikkonen
Kimmo Kaski

Helsinki University of Technology
Laboratory of Computational Engineering
P.O. Box 9400, FIN-02015 HUT
Finland

Keywords: Recurrent Self-Organizing Map, Temporal sequence processing

Abstract

Recurrent Self-Organizing Map (RSOM) is studied in temporal sequence processing. RSOM includes a recurrent difference vector in each unit of the map, which allows storing temporal context from consecutive input vectors fed to the map. RSOM is a modification of the Temporal Kohonen Map (TKM). It is shown that RSOM learns a correct mapping from temporal sequences of a simple synthetic data, while TKM fails to learn this mapping. In addition, two case studies are presented, in which RSOM is applied to EEG based epileptic activity detection and to time series prediction with local models. Results suggest that RSOM can be efficiently used in temporal sequence processing.

1 Introduction

Temporal sequence processing (TSP) is a research area having applications in diverse fields varying from weather forecasting to time series prediction, speech recognition and remote sensing. In order to construct a model for a process, data is gathered by measuring values of certain variables sequentially in time. Usually data is incomplete and includes noise. The goal for model building is to reveal the underlying process from data. Model is estimated by using statistical methods to find regularities and nonlinear dependencies that exist in the data. Usually the model that predicts the future of the process most accurately is considered to be the best model.

Several computational techniques have been proposed to gain more insight into processes and phenomena that include temporal information. Statistical methods based on linear (e.g. AR and ARMA)

and non-linear (e.g. NARMAX and MARS) have been effectively used in many applications [3]. Recently neural networks have gained a lot of interest in TSP due to their ability to learn effectively nonlinear dependencies from large volume of possibly noisy data with a learning algorithm. While many architectures e.g. multilayer perceptron (MLP) and radial basis function network (RBF) have been proved to be universal function approximators, this does not necessarily imply their usability in TSP.

Traditional way of using neural networks in TSP is to convert the temporal sequence into concatenated vector via a tapped delay line, and to feed the resulting vector as an input to a network [11]. This time-delay neural network approach, however, has its well know drawbacks, one of the most serious ones of being the difficulty to determine the proper length for the delay line. Therefore a number of dynamic neural networks models have been designed for TSP to capture inherently the essential context of the temporal sequence without the need of external time delay mechanics. In these models learning equations are often described by differential or difference equations and the interconnections between the network units may include a set of feedback connections, i.e., the networks are recurrent in nature (see [11, 14]).

Most recurrent neural networks are trained via supervised learning rules. Only quite rare unsupervised neural networks models have been proposed for TSP, although, it can be argued that in temporal sequence analysis unsupervised neural networks could reveal useful information from the temporal sequences at hand in analogy to unsupervised neural networks' reported power in cluster analysis, dimensionality reduction and visualization of their 'static'

input spaces. Moreover, in many TSP applications unsupervised learning could utilize more effectively the available temporal data than supervised learning methods, because no preclassification or prelabeling of the input data is needed. Based on the above the needs for the unsupervised learning methods in TSP are immense.

Temporal Kohonen Map (TKM) [1] is one interesting unsupervised approach for TSP derived from the Kohonen's Self-Organizing Map [6, 7] algorithm. In the TKM the involvement of the earlier input vectors in each unit is represented by using a recursive difference equation which defines the current unit activity as a function of the previous activations and the current input vector. Recurrent Self-Organizing Map (RSOM) proposed originally in [16] can be presented as an enhancement for the TKM algorithm. In brief RSOM defines a difference vector for each unit of the map which is used for selecting the best matching unit and also for adaptation of weights of the map. Difference vector captures the magnitude and direction of the error in the weight vectors and allows learning temporal context. Weight update is similar to the SOM algorithm—except that weight vectors are moved towards recursive linear sum of past difference vectors and the current input vector.

The rest of the paper is organized as follows. The RSOM algorithm is described in detail in section 2. Classification of temporal sequences, clustering of EEG patterns and time series prediction are considered in section 3. Finally some conclusions are made.

2 Recurrent Self-Organizing Map

We present as an extension to the Self-Organizing Map the Recurrent Self-Organizing Map (RSOM) [17, 8] that allows storing certain information from the past input vectors. The information is stored in the form of difference vectors in the map units. The mapping that is formed during training has the topology preservation characteristic of the SOM.

The Self-Organizing Map (SOM) [6] is a vector quantization method with topology preservation when the dimension of the map matches the true dimension of the input space. In brief topology preservation means that input patterns close in the input space are mapped to units close on the SOM lattice, where the units are organized into a regular N -dimensional grid. Furthermore units close on the SOM are close in the input space. Topology preservation is achieved with the introduction of topological neighborhood that connects the units of the SOM with a neighborhood function, h .

The training algorithm of the SOM is based on unsupervised learning, where one sample, the input vector $x(n)$ from the input space V_I , is selected randomly and compared against the weight vector w_i of the unit i the map space V_M . The best matching unit b to given input pattern $x(n)$ is selected using some metric based criterion, such as

$$\|x(n) - w_b(n)\| = \min_{i \in V_M} \{\|x - w_i(n)\|\}, \quad (1)$$

where $\|\cdot\|$ denote the Euclidean vector norm. Initially all weight vectors are set randomly to their initial positions in the input space. During the learning phase the weights in the map are updated towards the given input pattern $x(n)$ according to

$$w_i(n+1) = w_i(n) + \gamma(n)h_{ib}(n)(x(n) - w_i(n)), \quad (2)$$

where $i \in V_M$ and $\gamma(n)$, $0 \leq \gamma(n) \leq 1$, is a scalar valued adaptation gain. The neighborhood function, $h_{ib}(n)$, gives the excitation of unit i when the best matching unit is b . A typical choice for h_{ib} is a Gaussian function $h_{ib}(n) = \exp(-\|r_i - r_b\|^2/\sigma(n)^2)$, where σ controls the width of the function and r_i , r_b are the N -dimensional SOM index vectors of the unit i and the best matching unit b . During learning the function h_{ib} normally approaches delta function, i.e., σ slowly approaches zero as training progresses. When good quantization is desired the map has to be trained with only b in h_{ib} once the map has organized. During this quantization stage the gain has to be sufficiently small to avoid losing the map order, how small exactly varies from case to case. In order to guarantee convergence of the algorithm, the gain $\gamma(n)$ should decrease as a function of time or training steps according to conditions [13]:

$$\begin{aligned} \lim_{t \rightarrow \infty} \int_0^t \gamma(t') dt' &= \infty, \\ \lim_{t \rightarrow \infty} \int_0^t (\gamma(t'))^2 dt' &= C, C < \infty. \end{aligned} \quad (3)$$

If the map is trained properly, i.e the gain and the neighborhood functions are properly decreased over training a mapping is formed, where weight vectors specify centers of clusters satisfying the vector quantization criterion:

$$E = \min \left\{ \sum_{j=1}^M \|x_j - w_{b(x_j)}\| \right\}, \quad (4)$$

where we seek to minimize the sum squared distance E of all input patterns, x_j , $j = 1 \dots M$, to the respective best matching units with weight vectors $w_{b(x_j)}$.

Furthermore [2] relates the point density function, $p(w)$, of the weight vectors with the point density

function, $p(x)$, of the sampled underlying distribution in the V_I . Density function $p(w)$ is a better approximation of the underlying $p(x)$ than the approximation $p(x) \frac{d}{d+2}$. Here d is the dimension of the V_I , achieved with a random quantizer.

Chappell and Taylor [1] proposed a modification to the original SOM. This modification, being named Temporal Kohonen Map, is not only capable of separating different input patterns but is also capable of giving context to patterns appearing in sequences. Next we will describe Temporal Kohonen Map in more detail.

2.1 Temporal Kohonen Map

Temporal Kohonen Map (TKM) differs from the the SOM only in its outputs. The outputs of the normal SOM are reset to zero after presenting each input pattern and selecting the best matching unit with the typical winner take all strategy or making other use of the unit output values, hence the map is sensitive only to the last input pattern. In the TKM the sharp outputs are replaced with leaky integrator outputs which, once activated, gradually lose their activity.

The modeling of the outputs in the TKM is close to the behavior of natural neurons, which retain an electrical potential on their membranes with decay. In the TKM this decay is modeled with the difference equation:

$$V_i(n) = dV_i(n-1) - (1/2)\|x(n) - w_i(n)\|^2, \quad (5)$$

where $0 < d < 1$ can be viewed as a time constant, $V_i(n)$ is the activation of the unit i at step n , $w_i(n)$ is the reference or the weight vector in the unit i and $x(n)$ is the input pattern. Now the best matching unit b is the unit with maximum activity. Equation Eq. 5 has the following general solution:

$$V_i(n) = -(1/2) \sum_{k=0}^{n-1} d^k \|x(n-k) - w_i(n-k)\|^2 + d^n V_i(0), \quad (6)$$

where the involvement of the earlier inputs is explicit. Further analysis of Eq. 6 shows how the optimal weight vectors in the vector quantization sense can be solved explicitly when n is assumed to be sufficiently large to render the last residual term corresponding to initial activity insignificant. The analysis, when w_i is assumed constant, goes as follows:

$$\frac{\partial V(n)}{\partial w} = - \sum_{k=0}^{n-1} d^k (x(n-k) - w). \quad (7)$$

Now, when w is optimal in the vector quantization sense (Eq. 4) the derivative in Eq. 7 is zero as this

minimizes the sum in Eq. 6. Hence substituting the left hand side of Eq. 7 with 0 yields:

$$0 = - \sum_{k=0}^{n-1} d^k (x(k) - w),$$

$$w = \frac{\sum_{k=0}^{n-1} d^k (x(n-k))}{\sum_{k=0}^{n-1} d^k}. \quad (8)$$

The result shows how the optimal weight vectors in the vector quantization sense are linear combinations of the input patterns. Since the TKM is trained with the normal SOM training rule, it attempts to minimize the normal vector quantization criterion in Eq. 4, which is other than the criterion suggested by Eq. 8. As a consequence it appears that it may be possible to properly train a TKM only for relatively simple input spaces.

2.2 Modified TKM: RSOM

Some of the problems of the original TKM have a convenient solution in simply moving the leaky integrators from the unit outputs into the inputs. This gives rise to the modified TKM called Recurrent Self-Organizing Map, RSOM.

Moving the leaky integrators from the outputs into the inputs yields

$$y_i(n) = (1 - \alpha)y_i(n-1) + \alpha(x(n) - w_i(n)), \quad (9)$$

for the temporally leaked difference vector at each map unit. Above $0 < \alpha \leq 1$ is the leaking coefficient analogous to d in the TKM, $y_i(n)$ is the leaked difference vector while $x(n)$ and $w_i(n)$ have their previous meanings. Schematic picture of an RSOM unit is shown in Fig. 1.

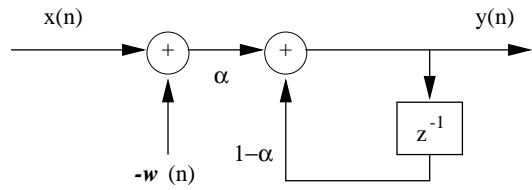


Figure 1: Schematic picture of an RSOM unit which acts as a recurrent filter.

Large α corresponds to short memory while small values of α correspond to long memory and slow decay of activation. In the extremes of α RSOM behaves like a normal SOM ($\alpha = 1$) while in the other extreme all units tend to the mean of the input data.

Eq. 9 can be written in a familiar form by replacing $x'_i(n) = x(n) - w_i(n)$, yielding:

$$y_i(n) = (1 - \alpha)y_i(n-1) + \alpha x'_i(n); \quad (10)$$

which describes an exponentially weighted linear IIR filter with the impulse response

$h(k) = \alpha(1 - \alpha)^k$, $k \geq 0$. For further analysis of the Eq. 10, see e.g. [12].

Since the feedback quantity in RSOM is a vector instead of a scalar it also captures the direction of the error which can be exploited in weight update when training the map. The best matching unit b at step n is now searched by

$$y_b = \min_i \{ \|y_i(n)\| \}, \quad (11)$$

where $i \in V_M$. Then the map is trained with a slightly modified Hebbian training rule given in Eq. 2 where the difference vector, $(x(n) - w_i(n))$ is replaced with y_i . Thus the unit is moved toward the linear combination of the sequence of input patterns captured in y_i .

Repeating the mathematical analysis on RSOM earlier done with the TKM in Eqs. 7 and 8 yields:

$$y(n) = \alpha \sum_{k=1}^n (1 - \alpha)^{(n-k)} (x(k) - w).$$

The square of the norm of $y(n)$ is

$$\|y(n)\|^2 = \alpha \sum_{k=1}^n (1 - \alpha)^{(n-k)} (x(k) - w)^T (x(k) - w). \quad (12)$$

Optimizing the vector quantization criterion given in general form in Eq. 4 with respect to y yields the following condition:

$$\frac{\partial \|y(n)\|^2}{\partial w} = -2\alpha \sum_{k=1}^n (1 - \alpha)^{(n-k)} (x(k) - w) = 0, \quad (13)$$

when w is optimal. The optimal w can be analytically solved:

$$w = \frac{\sum_{k=1}^n (1 - \alpha)^{(n-k)} x(k)}{\sum_{k=1}^n (1 - \alpha)^k}. \quad (14)$$

From Eq. 14 one immediately observes how the optimal w 's are linear combinations of the x 's. Note how this result is essentially identical with the result in Eq. 8 so it might seem that the algorithms are essentially the same. However since RSOM is trained with the y 's it seeks to minimize the quantization criterion suggested by Eq. 13 while the TKM seeks to minimize the normal vector quantization criterion in Eq. 4. Nevertheless the resolution of RSOM is limited to the linear combinations of the input patterns with different responses to the operator in the unit inputs. If a more sophisticated memory is required one has to resort to multilayer structures like those presented in [5].

3 Case Studies

Three different cases are presented to evaluate the usability of RSOM in temporal sequence processing. First case is a synthetic data which consist of

random sequences of symbols with a limited alphabet and additive noise. In second case RSOM is used to cluster feature vectors extracted from EEG to classify the signal as being either normal or containing epileptic activity. In third case local models are used with RSOM for time series prediction. Laser data set which is publicly available is used and results are compared with multilayer perceptron (MLP) and linear AR models.

3.1 Temporal Sequence Classification

This first synthetic case aims to underline the differences of the TKM and the RSOM. Both RSOM and TKM were trained with five one-dimensional input patterns, 1, 6, 11, 16, 21, with additive approximately Gaussian noise.

The optimal weights for each α or d can be directly computed using either Eq.s 8 or 14 and the distribution of the weights can be used to choose the optimal α when the goal is known. In this case we seek to distinguish sequences of input patterns. A good heuristic criterion for weights is then that the weights are to be as evenly distributed across the entire input space as possible. Using this heuristic criterion we arrived at optimal $\alpha = 0.8$ and accordingly optimal $d = 0.2$.

Six different maps, three TKM:s and three RSOM:s, with 25 units and $\alpha \in \{0.7, 0.8, 0.9\}$ were trained to demonstrate the behavior of the training algorithms against analytically solved optimal weights for each α . The maps were trained for 2000 iterations per unit to give the maps ample time to converge. Such a map should, when properly trained, be able to distinguish all 25 ordered pairs of inputs shown to the map as a sequence of length two. The resulting weights with the optimal weights are shown in Figures 2, 3 and 4. The optimal weights are marked with small circles "o", the weights of the RSOM are marked with plus signs "+" and the corresponding weights of the TKM are marked with small x:s "x".

In the case of RSOM the units are updated explicitly toward the linear combinations of input patterns thus the map seeks to learn the optimal weights.

However, with the TKM the situation is more complicated. Basically the TKM has two mechanisms to retain contextual information. Consider the example in this section. If the input activity moves from 1 to 21, for a short period of time some unit u_i , close to 1, remains the best matching unit and is replaced by a unit, u_j , near 21 when the activity of u_i is decayed below the raising activity of u_j . As a consequence u_i with its neighbors are trained toward 21 for a brief moment. The motivation behind the TKM is to learn temporal context from the past but instead of learning to distinguish where the activity came from it seems to learn where the activity

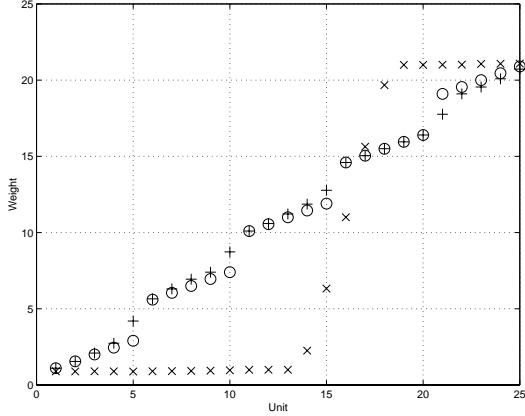


Figure 2: The weights with $\alpha = 0.9$. Optimal weights with “o”, RSOM weights with “+” and TKM weights with “x”.

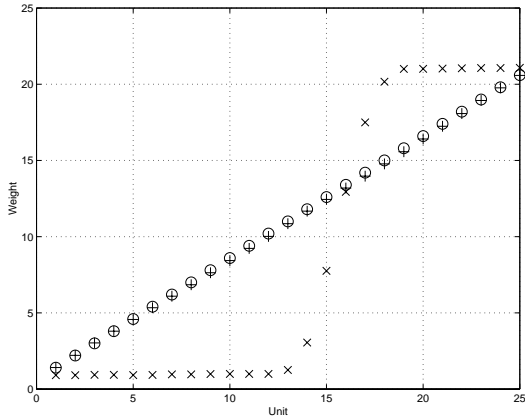


Figure 3: The weights with $\alpha = 0.8$. Optimal weights with “o”, RSOM weights with “+” and TKM weights with “x”. These weights are computed with the optimal α according to our heuristic criterion.

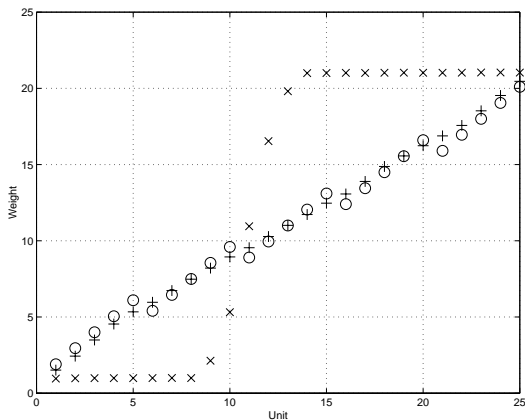


Figure 4: The weights with $\alpha = 0.7$. Optimal weights with “o”, RSOM weights with “+” and TKM weights with “x”.

went.

The other mechanism in the TKM that preserves contextual information is topological neighborhood, which is inherent to all SOM algorithms. Consider a situation of three separate regions of activity on a straight line and a map of five units. Now as the map is trained three of the units converge at the centroids of the regions of activity while the remaining two are left dangling in the zero probability regions separating the regions of activity. These dangling units may then be capable of separating some sequences. This is not, however, an explicit method of capturing past.

A distinctive feature in the trained TKM:s is the concentration of the units near the edges of the input manifold. Brief study of the properties of the training rule showed how this phenomena is almost unavoidable.

3.2 Clustering of EEG Patterns

The second case is related to EEG spectral feature clustering for epileptic activity detection [15]. EEG is an important clinical tool for diagnosing, monitoring and managing neurological disorders related to epilepsy. Since epileptiform activity can be noticed in EEG as a clearly distinguishable transient waveforms, wavelets have been used efficiently to extract suitable features for epilepsy detection.

The sampling rate of the EEG data used in the test was 200 Hz. For spectral feature extraction at time t a feature extraction window W^t of 256 samples was collected from the original EEG sequence S_{EEG} as follows: $W^t(i) = S_{EEG}(t - 127 + i)$, $i = 0, \dots, 255$. The wavelet transform of a continuous signal $g(t)$ with a wavelet $\Psi((t - b)/a)$ is given as: $C_{a,b}^t = \int_{-\infty}^{\infty} g(t)\Psi((t - b)/a) dt$, where a is the scaling and b is the dilation factor of the wavelet Ψ . In our case two Daubechies' mother wavelets, Daub₄ and Daub₁₂, were employed and the discrete wavelet transform for each window W^t was done using Mallat's “with holes” [10] algorithm. Finally, total of sixteen energy features f^t were determined for each W^t by computing the squared sum of both Daub₄ and Daub₁₂ wavelet coefficients at each scale.

The extracted EEG features were first clustered by four SOMs of 3×3 , 5×5 , 9×9 and 17×17 units in two dimensional lattice. The Luttrell's method [9] was used in training. The training data contained total of 150987 16-dimensional feature vectors, among which 5430 patterns correspond to epileptic activity. After training each unit of the map was labeled according to the plurality rule to belong either to “normal” or “epileptic” activity, and the SOMs were used as classifiers to evaluate the discrimination potentials of the feature clusters. Note that for labeling the number of epileptic activity samples mapped into each unit i was multiplied by the factor $145557/5430$ for equal weighting of both class-

es. Table 1. shows the confusion matrices of the four SOM classifiers; the diagonal entries give the correctly classified samples. These results were compared to the RSOMs of same sizes. The α parameter was set to value 0.6, and in each learning cycle 5 past feature values f^{t_c-64} , f^{t_c-48} , f^{t_c-32} , f^{t_c-16} , and f^{t_c} were shown to the map with randomly selected time t_c . The RSOM maps were taught with the Luttrell's approach [9]. When the labeled RSOMs were employed to classify training samples, the results were better than those of normal SOMs with the same number of units (see Table 1.). These clustering results suggest that the use of the context in the EEG based epileptic activity detection might improve the analysing/classification results, and the RSOM may be a valuable tool for the purpose.

Table 1: SOM and RSOM confusion matrices obtained in the clustering of EEG spectral features (see text for details)

# of units	SOM		RSOM	
3×3	126966	18591	129269	16224
	405	5025	321	5109
5×5	131794	13763	133389	12104
	443	4987	380	5050
9×9	133820	11737	134660	10833
	394	5036	317	5113
17×17	134465	11092	135935	9558
	338	5092	284	5146

3.3 Time Series Prediction

Third case is time series prediction with RSOM and local linear models. Laser time series [18] (Fig 5.) consists of measurements of the intensity of an infrared laser in a chaotic state. The data is available from an anonymous ftp server ¹. From the beginning of the series first 2000 samples were used for training, and the rest 1000 samples were used for testing. Both series were scaled between [-1,1]. The prediction task was one-step prediction.

Learning algorithm of RSOM for time series prediction is implemented as follows. The map is presented an *episode* of consecutive input vectors starting from a random point in the data. The number of vectors belonging to the episode is dependent on the leaking coefficient α used in the units. At the end of the episode, the impulse response $h(k)$ of the recurrent filter (see Fig. 1) is below 5 % of the initial value. The best matching unit is selected at the end of the episode based on the norm of the difference vector. The updating of the vector and its neighbors is carried out as in Eq. 2. After the updating, all difference vectors are set to zero, and a new random

¹ftp://ftp.cs.colorado.edu/pub/Time-Series/SantaFe/ containing files A.dat (first 1000 samples) and A.cont (as a continuation to A.dat 10000 samples)

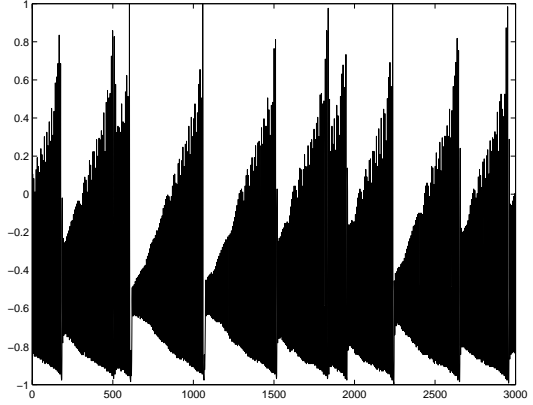


Figure 5: Laser time series.

starting point from the series is selected. The above scenario is repeated until the mapping has formed. Training set is then divided into local data sets according to the best matching unit on the map, and associated local models are estimated using these data sets. In prediction the best matching unit of RSOM is searched for each input vector. A local model that is associated with the best matching unit is then selected to be used in the prediction task at that time.

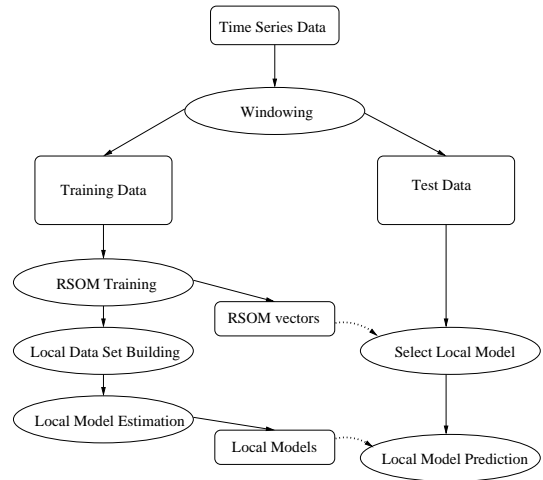


Figure 6: Construction of the local models.

Figure 6. shows the procedure for building the RSOM models and evaluating their prediction abilities with testing data [8]. Time series is divided to training and testing data. Input vectors to RSOM are formed by windowing the time series. Free parameters during training for RSOM include input vector length p , time step between consecutive input vectors s , number of units n_u and the leaking coefficient α of the units in the map giving rise to model $RSOM(p, s, n_u, \alpha)$. Parameters were varied as $n_u \in \{5, 9, 13\}$, $s \in \{1, 3, 5\}$, $p \in \{3, 5, 7\}$ and $\alpha \in \{1.0, 0.95, 0.78, 0.73, 0.625, 0.45, 0.40, 0.35\}$

corresponding to episode lengths 1 ... 8. Local linear regression models were estimated using the least squares algorithm in MATLAB 5 statistics toolbox using the local data sets generated with RSOM.

For model selection purposes 4-fold cross-validation [4] was used. The best model according to cross-validation is trained again with the whole training data. This model is then used to predict the test data set that has not been presented to the model before. The same cross-validation scheme was used also for MLP and AR models.

The MLP network was trained with Levenberg-Marquardt learning algorithm implemented with MATLAB 5 neural networks toolbox. An $MLP(p,s,q)$ network with one hidden layer, p inputs and q hidden units was used. Variation of parameters p and s were chosen to be the same as in RSOM models, while q was varied as $q \in \{3, 5, 7, 9\}$.

$AR(p)$ models with p inputs were estimated with MATLAB 5 using the least-squares algorithm. The order of the AR model was varied as $p \in \{1, \dots, 50\}$. Results of the AR model serve as an example of the accuracy of a global linear model in the current tasks.

The sum-squared errors gained for one-step prediction task are shown in Table 2. The laser series is highly nonlinear and thus the errors gained with $AR(12)$ model are considerably higher than for other models. The series is also stationary and almost noiseless, which explains the accuracy of the $MLP(9,1,7)$ model predictions. In this case $RSOM(3,3,13,0.73)$ gives results that are better than with AR model but worse than with MLP model.

Table 2: One-step Prediction Errors for Laser Time Series.

	CV Error	Test Error
RSOM(3,3,13,0.73)	14.6995	7.3894
MLP(9,1,7)	4.9574	0.9997
AR(12)	69.8093	29.8226

4 Conclusions

Recurrent Self-Organizing Map architecture and learning algorithm has been presented. Studied cases show the potentials of RSOM in temporal sequence processing. Results with synthetic data show that RSOM learns a correct mapping which agrees with vector quantization criterion. In EEG case feature vectors were clustered with SOM and RSOM. Temporal context captured by RSOM provided better results when a simple prularity rule was used for classification. Results in prediction case are not the best possible with RSOM, since the search space of the free parameters of the model was quite small.

An important property of RSOM is its visualization ability. RSOM can be used as a visualization tool since it has built-in property of SOM to construct topological maps from data. For instance, in the case of time series prediction a two-dimensional map could be used to visualize state changes of the process using the location of the best matching unit in the map as function of time.

Unsupervised learning of the temporal context is another attractive property of RSOM. It allows building models using large amount of data with only a little *a priori* knowledge, e.g. in classification there is no need for pre-labeled cases. This property also allows using RSOM as an explorative tool to find statistical temporal dependencies from the process under consideration.

In this paper we presented RSOM that has the same feedback structure in all units. It is possible, however, to allow the units of RSOM to have different recurrent structures. This kind of architecture could yield topological maps of different temporal contexts in the data. It is our intention to study such extensions of RSOM in the near future.

Acknowledgments

We wish to thank Dr.Tech. Alpo Värri from Tampere University of Technology, Finland, for generously providing us the scored EEG recordings of the ANNDEE project. This study has been funded by *Academy of Finland* and *Technology Development Centre (TEKES)*.

References

- [1] G.J. Chappell and J.G. Taylor. The temporal Kohonen map. *Neural Networks*, 6:441–445, 1993.
- [2] M. Cottrell. Theoretical aspects of the SOM algorithm. In *Proc. of Workshop on Self-Organizing Maps*, pages 246–267. Helsinki University of Technology, 1997.
- [3] N. Gershenfeld and A. Weigend. The future of time series: Learning and understanding. In A. Weigend and N. Gershenfeld, editors, *Time Series Prediction: Forecasting the Future and Understanding the Past*, pages 1–70. Addison-Wesley, 1993.
- [4] L. Holmström, P. Koistinen, J. Laaksonen, and E. Oja. Neural and statistical classifiers—taxonomy and two case studies. *IEEE Trans. Neural Networks*, 8(1):5–17, 1997.

- [5] J. Kangas. *On the Analysis of Pattern Sequences by Self-Organizing Maps*. PhD thesis, Helsinki University of Technology, May 1994.
- [6] T. Kohonen. *Self-Organization and Associative Memory*. Springer-Verlag, Berlin, Heidelberg, 1989.
- [7] T. Kohonen. *Self-Organizing Maps*. Springer-Verlag, 1995.
- [8] T. Koskela, M. Varsta, J. Heikkonen, and K. Kaski. Time series prediction using recurrent SOM with local linear models. *Int. J. of Knowledge-Based Intelligent Engineering Systems*, in press. Available as research reports B15, Helsinki University of Technology, Lab. of Computational Engineering, 1997.
- [9] S. Luttrell. Image compression using a multi-layer neural network. *Pattern Recognition Letters*, 10:1–7, 1989.
- [10] S.G. Mallat. A theory of multiresolution signal decomposition: The wavelet representation. *IEEE Trans. Pattern Anal. Mach. Intell. (PAMI)*, 11:674–693, 1989.
- [11] M. Mozer. Neural net architectures for temporal sequence processing. In A. Weigend and N. Gershenfeld, editors, *Time Series Prediction: Forecasting the Future and Understanding the Past*, pages 243–264. Addison-Wesley, 1993.
- [12] J.G. Proakis and D.G. Manolakis. *Digital Signal Processing: Principles, Algorithms, and Applications*. MacMillan Publishing Company, 1992.
- [13] H. Ritter and K. Schulten. Convergence properties of Kohonen's topology preserving maps: Fluctuations, stability and dimension selection. *Biological Cybernetics*, 60:59–71, 1988.
- [14] A. Tsoi and A. Back. Locally recurrent globally feedforward networks: A critical review of architectures. *IEEE Transactions on Neural Networks*, 5(2):229–239, 1994.
- [15] M. Varsta, J. Heikkonen, and J. del R. Millán. Epileptic activity detection in EEG with neural networks. In *Proc. Int. Conf. on Engineering Applications of Neural Networks, EANN'97*, pages 179–186. Royal Institute of Technology, Stockholm, 1997.
- [16] M. Varsta, J. Heikkonen, and J. Del Ruiz Millán. Context learning with the self-organizing map. In *Proc. of Workshop on Self-Organizing Maps*, pages 197–202. Helsinki University of Technology, 1997.
- [17] M. Varsta, J. Del R. Millán, and J. Heikkonen. A recurrent Self-Organizing Map for temporal sequence processing. In *Proc. 7th Int. Conf. Artificial Neural Networks, ICANN'97*, pages 421–426. Springer-Verlag, 1997.
- [18] A. Weigend and N. Gershenfeld, editors. *Time Series Prediction: Forecasting the Future and Understanding the Past*. Addison-Wesley, 1993.