# Using the SOM and Local Models in Time-Series Prediction

Juha Vesanto

Helsinki University of Technology
Laboratory of Computer and Information Science
Rakentajanaukio 2 C, FIN-02150 Espoo, Finland
e-mail Juha.Vesanto@hut.fi

**Abstract**

In this paper we test the Self-Organizing Map (SOM) on the problem of predicting chaotic time-series (specifically Mackey-Glass series) with local linear models defined separately for each of the prototype vectors of the SOM. We see that the method achieves good results. This together with the capabilities of the SOM make it a valuable tool in exploratory data mining.

## 1 Introduction

The problem of time-series prediction is one of high practical importance. A traditional way to approach the problem is to estimate the underlying function globally. In this kind of approach neural methods like the Multi-Layer Perceptron (MLP) and Radial Basis Functions (RBF) are popular. In the last decade, however, local models have been a source of much interest because in many cases they give better results than global models [1]. This is especially true if the function characteristics vary throughout the feature space.

Local models are usually based on nearest-neighbour methods. The pure k-Nearest-Neighbours (kNN) -methods are effective, but to perform well they need a fair amount of example vectors which implies a large memory requirement. To overcome the problem we need to select a presentable subset of prototype vectors, i.e. to quantize the feature space. Local models themselves can be constructed in various ways ranging from using the best example vector as such to splines and small MLPs. Usually, local models are kept simple such as weighted averages of the example vectors or, as in this paper, linear regression models.

The Self-Organizing Map (SOM) [2] is an efficient method for vector quantization. In the usual case where the dimension of the map grid is lower than the inherent dimension of the feature space [3] the quantization process of the SOM is somewhat disturbed by the fact that in addition to the quantization, the SOM also tries to preserve the topology of the data space on the lower-dimensional grid. Other quantization methods, like *K-means* or neural gas [4] do not suffer from this limitation, but using the SOM offers other benefits: the SOM is an effective tool for exploratory data analysis [5] and it has prominent visualization capabilities.

Previously the SOM has been used in chaotic time-series prediction e.g. by Walter *et al.* [6], Der and Herrman [7], and Principe [8].

## 2 Methodology and Implementation

In time-series prediction, the time-series is usually first embedded in a state space using delay coordinates:

$$\mathbf{x}(n) = [x(n), x(n-\tau), ..., x(n-(N-1)\tau)], \tag{1}$$

where $x(n)$ is the value of the time-series at time $n$, $\tau$ a suitable time delay and $N$ the order of the embedding. This embedded vector is then used to predict the next value of the series $x(n+\tau)$.

A generalized scheme for constructing and testing the local models is depicted in figure 1. The data is embedded and then divided into the training and the testing set. Based on the training set, the embedded data space is quantized. Local data sets are then constructed for each of the prototype vectors: the local data set of a prototype vector is formed of those data vectors to which it is closest (i.e. of data vectors falling to the prototype vector's Voronoi plate). If a local data set of a prototype vector is considered too small, it can be augmented from the data sets of similar, close lying prototype vectors. At the last step, local models are constructed based on the local data sets.

The performance of the model is tested by taking an embedded vector from the testing set, selecting the best prototype vector and predicting the next value using the corresponding local model. The time-series can be predicted further by inserting the predicted value into the test vector and iterating the process.

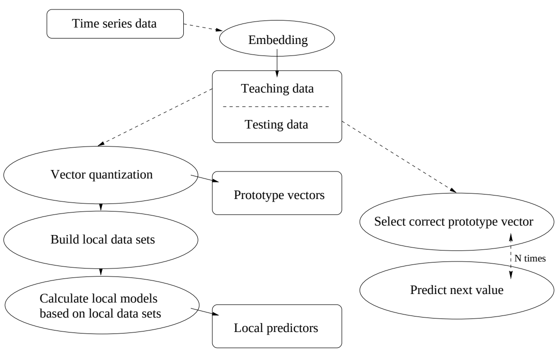


Figure 1: *A generalized scheme for constructing and testing the local models.*

## 2.1 Implementation

In this work, the SOM was used to quantize an embedded vector space of one dimension greater than the one in equation 1, namely one consisting of the next value of the time-series (the target value) and the embedded vector:

$$\mathbf{y}(n) = [\gamma x(n+\tau), \mathbf{x}(n)]. \tag{2}$$

Scaling the first component by $\gamma$, its variance with respect to other vector components can be controlled. In training the SOM, the variables with largest variances typically dominate the learning process. The target value was included in the vector because it provides extra information which may improve the results.

The SOM was trained using the SOM_PAK program package [9]. The map was initialized along a two-dimensional subspace spanned by the two principal eigenvectors of the input data vectors, and the SOM was trained for 50 epochs. Initial neighbourhood radius was approximately half of the sidelength of the map, initial value for the learning coefficient was 0.3 and it was decreased proportional to inverse of time. The map topology was rectangular. The rest of the method was implemented in `matlab` script, so the running times are not directly comparable to other studies.

The local data sets were constructed by searching the best matching prototype vector for each of the embedded data vectors. In searching the best match the target value in data vectors $\mathbf{y}$ (equation 2) was left out. If the number of items in a local data set was less than a predefined value, the local data set was augmented by the data set of the closest prototype vector. If there still was not enough items in the data set, the data set of the second closest prototype vector was also included, and so on.

Linear regression was used to build the local models:

$$\hat{x}(n+1) = a_N + \sum_{i=0}^{N-1} a_i x(n-i), \tag{3}$$

where $\hat{x}(n+1)$ is the prediction, $x(n-i)$ are the components of the embedded data vectors and $a_i$ the parameters of the model. To ensure the stability of the regression model in least squares sense, at least $N+1$ data vectors should be used in the estimation. In this work, the minimum local data set size was set to 12.

# 3 Experiments

The time-series that was used in testing the method, the Mackey-Glass system [10], is a time-delay differential system of the form:

$$\frac{dx}{dt} = \beta x(t) + \frac{\alpha x(t-\delta)}{1 + x(t-\delta)^{10}}, \tag{4}$$

where $x(t)$ is the value of time-series at time $t$. The system is chaotic for $\delta > 16.8$. The data set was constructed using parameter values $\beta = -0.1, \alpha = 0.2$ and $\delta = 17$. Separate and independent training (3001 samples) and testing sets (501 samples) were generated. The embedded data vectors consisted of four values of the time-series: $\mathbf{x}(n) = [x(n), x(n-6), x(n-12), x(n-18)]$ and the predicted value was $x(n+6)$ ($N = 4, \tau = 6$ in equation 1). For each testing sample the prediction scheme was iterated 15 times so that the last value corresponded to predicting $x(n+90)$ [1].

The results in this section are given as the root mean squared error normalized by the variance of the original time-series (NRMS [2] ).

## 3.1 Results

The performance of the proposed method was first tested on various sized SOMs. The target value (first component in equation 2) was left unmodified ($\gamma = 1$). The results are depicted in table 1. It can be seen that the performance of the method increases as a function of the size of the SOM. However, when the number of prototype vectors exceeds the number of training samples the error starts to rise again. It should also be noted that if we use a very large amount of prototype vectors, we lose the compactness that is achieved by the quantization process and prediction process becomes computationally heavier. Therefore the size $35 \times 35$ was selected for further experiments.

Eight $35 \times 35$ sized maps were trained with different scalings of the target value. The results are listed in table 2. It seems that taking the target value into the quantization process has some positive effect, if it's scaled appropriately. In further experiments a scaling factor of $\gamma = 0.1$ was used.

Finally the method was tested on varying training set sizes. For this purpose a big training set of 10001 samples was prepared. The maps were trained for approximately equal lengths (150000 samples). To test the limits of the proposed method a $100 \times 100$ SOM was trained using the 10001 samples data set. The results are depicted in table 3.

[1] It should be noted that because of the embedding and the length of prediction, the number of samples in both training and testing sets was decreased by a number of samples: the training sets fall 24 samples and the test sets 114 samples short of the indicated numbers.

[2] $error = \sqrt{\sum_{i=0}^{N-1} (\hat{x(i)} - x(i))^2 / N\sigma_x^2}$, where $\hat{x(i)}$ is the predicted value, $x(i)$ is the correct value, $N$ is the number of samples and $\sigma_x^2$ is the variance of the time-series.

Table 1: *Normalized root mean squared errors for various sized SOMs. Second row gives NRMS for first prediction step ($x(n+6)$) and the third row the NRMS for the 14th step ($x(n+84)$).*

| Mapsize | x(n+6) | x(n+84) |
|---|---|---|
| 5x5 | 0.030 | 0.13 |
| 10x10 | 0.013 | 0.060 |
| 20x20 | 0.010 | 0.14 |
| 35x35 | 0.010 | 0.032 |
| 50x50 | 0.0036 | 0.016 |
| 75x75 | 0.0073 | 0.021 |

Table 2: *Normalized root mean squared errors for various SOMs (grid size $35 \times 35$) with different scalings given for the first component (the target value). Second row gives NRMS for first prediction step ($x(n+6)$) and the third row the NRMS for the 14th step ($x(n+84)$).*

| $\gamma$ | x(n+6) | x(n+84) |
|---|---|---|
| 0 | 0.0065 | 0.022 |
| 0.1 | 0.0048 | 0.022 |
| 0.25 | 0.0050 | 0.029 |
| 0.5 | 0.015 | 0.21 |
| 1 | 0.010 | 0.032 |
| 2 | 0.017 | 0.070 |
| 4 | 0.011 | 0.058 |
| 10 | 0.022 | 0.087 |

## 3.2 Comparison with other methods

The Mackey-Glass time-series is a widely used benchmark data set. Typically the value $x(n+85)$, or $x(n+84)$ in algorithms that are used iteratively, is predicted. In the latter case, also the value $x(n+6)$ is usually predicted.

Deco and Obradovic used decorrelated hebbian learning (DHL) algorithm to predict the time-series at $x(n+85)$ [11]. Platt used a resource allocating network (RAN) [12] and Littman and Ritter direct cascade architecture with local linear map subnetworks (DCA+LLM) in the same task [13]. With training set of 500 samples the error was 0.043. Lapedes and Farber used standard backpropagation network with 500 training samples [14]. The prediction errors have been collected to table 4.

The proposed method compares well with these results. With training set of 1001 samples and $35 \times 35 = 1225$ prototype vectors the proposed method produced NRMS of 0.035 for $x(n+84)$. With only 100 prototype vectors, but with a larger data set (3001 points) a slightly worse result, 0.060, was achieved. NRMS of 0.022 was achieved with 3001 training samples and a $35 \times 35$ map.

Table 3: *Normalized root mean squared errors for SOMs trained with varying training set sizes. Second row gives NRMS for first prediction step ($x(n+6)$) and the third row the NRMS for the 14th step ($x(n+84)$).*

| Training set size | Map size | x(n+6) | x(n+84) |
|---|---|---|---|
| 501 | 35x35 | 0.033 | 0.12 |
| 1001 | 35x35 | 0.0091 | 0.035 |
| 3001 | 35x35 | 0.0048 | 0.022 |
| 10001 | 35x35 | 0.0024 | 0.012 |
| 10001 | 100x100 | 0.0014 | 0.0082 |

Table 4: *NRMS errors produced by various methods in predicting the values $x(n+6)$ and either $x(n+84)$ or $x(n+85)$ of the Mackey-Glass time-series ($\tau = 17$).*

| Method | Training set size | $x(n+6)$ | $x(n+84)$ or $x(n+85)$ |
|---|---|---|---|
| DHL | 1000 | | 0.056 |
| RAN | 3000 | | 0.054 |
| DCA+LLM | 500 | | 0.043 |
| Backprop | 500 | 0.02 | 0.06 |
| SOM (35x35) | 1001 | 0.0091 | 0.035 |
| SOM (35x35) | 3001 | 0.0048 | 0.022 |
| SOM (10x10) | 3001 | 0.013 | 0.060 |

Another widely used Mackey-Glass problem is otherwise similar, except for a bigger delay parameter ($\delta = 30$). For this problem, Principe used a time-delay neural network with a global feedback loop and achieved errors of 0.025 for $x(n+6)$ and $\approx 0.04$ for $x(n+84)$ using 500 training samples [15]. For a similar problem, the proposed method achieved prediction errors of 0.044 and 0.15 respectively with 3000 training samples. The problem is much harder and it may be that the local linear models are insufficient in flexibility. Figure 2 shows the MSE for each node of the SOM. It can be seen that most of the error comes from specific areas. A way to improve the results could be to use a more complex model, a higher-order polynomial or a small MLP, in the regions where the errors are high.

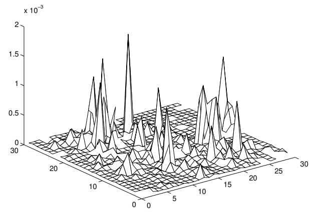

Figure 2: *Mean squared error in each node of a $30 \times 30$ SOM trained for the Mackey-Glass ($\tau = 30$) problem. The errors in the 10 biggest peaks corresponded to about 14% of the total sum of squared errors.*

# 4 Conclusions

The Self-Organizing Map coupled with local linear models is an efficient method for function approximation. In the case of the Mackey-Glass time-series it produces results that are directly comparable with or even better than those of other methods. Overfitting is not a significant problem with the proposed method if only some simple caution is kept in assigning the size of the SOM. The proposed method is also easy to modify to accommodate more complex problems by simply using more complex local models in areas where the linear model is insufficient.

While it can be argued that other quantization methods perform equally well or better than the SOM, they lack the other capabilities of the SOM. In this paper, it was shown that local linear models that can be "plugged-in" on top of a SOM are an addition to this set. It is the combination of these many abilities which make the SOM a valuable tool in exploratory data mining.

## Acknowledgments

This work has been carried out in the technology program "Adaptive and Intelligent Systems Applications" financed by the Technology Development Center of Finland (TEKES).

## References

[1] A. C. Singer, G. W. Wornell and A. V. Oppenheim. A Nonlinear Signal Modeling Paradigm. In *Proc. of ICASSP 1992*, 1992.

[2] T. Kohonen. *Self-Organizing Maps.* Springer-Verlag, 1995.

[3] J. Theiler. Estimating fractal dimension. In *Journal of the Optical Society of America*, vol. 7, no. 6, pp. 1055-1073, June 1990.

[4] T. M. Martinetz, S. G. Berkovich and K. J. Schulten. "Neural-Gas" Network for Vector Quantization and its Application to Time-Series Prediction. In *IEEE Transaction on Neural Networks*, 558–569, Vol. 4, No. 4, 1993.

[5] S. Kaski. *Data Exploration Using Self-Organizing Maps.* Acta Polytechnica Scandinavica, Mathematics, Computing and Management in Engineering Series, No. 82. Doctoral Thesis, Helsinki University of Technology, 1997.

[6] J. Walter, H. Ritter and K. Schulten. Non-linear Prediction with Self-organizing Maps. In *Proc. of th IJCNN 1990.*, 1990.

[7] R. Der and M. Herrmann. Nonlinear Chaos Control by Neural Nets. In *Proc. of the ICANN 94*, 1994.

[8] J. C. Principe and L. Wang. Non-Linear Time Series Modeling with Self-Organization Feature Maps. In *Proceedings of the 1995 IEEE Workshop. Neural Networks for Signal Processing V.*, 1995.

[9] T. Kohonen, J. Hynninen, J. Kangas and J. Laaksonen, SOM_PAK: The Self-Organizing Map Program Package. *Technical Report A31 at Helsinki University of Technology, Laboratory of Computer and Information Science*, 1996. Program package available via WWW at URL http://nucleus.hut.fi/nnrc/som_pak/.

[10] M. C. Mackey ja L. Glass. Oscillations and Chaos in Physiological Control Systems. In *Science*, 197–287, 1977.

[11] G. Deco and D. Obradovic. Decorrelated Hebbian Learning for Clustering and Function Approximation. In *Neural Computation*, volume 7, pages 338–348, 1995.

[12] J. Platt. Learning by Combining Memorization and Gradient Descent. In *Advances in Neural Information Processing Systems*, 1991.

[13] E. Littmann and H. Ritter. Learning and Generalization in Cascade Network Architectures. In *Neural Computation*, volume 8, pages 1521–1539, 1996.

[14] A. Lapedes and R. Farber. *Nonlinear Signal Processing Using Neural Networks: Prediction and System Modelling.* Tech. Report TR LA-UP-87-2662, Los Alamos National Laboratory, Los Alamos, 1987.

[15] J. C. Principe, J-M. Kuo. Non-linear Modelling of Chaotic Time Series with Neural Networks. In *Proc. of NIPS 7*, 1995.