ESANN'2006 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 26-28 April 2006, d-side publi., ISBN 2-930307-06-4.

LS-SVM Functional Network for Time Series Prediction

Tuomas Kärnä[1], Fabrice Rossi[2] and Amaury Lendasse[1]

Helsinki University of Technology - Neural Networks Research Center
P.O. Box 5400, FI-02015 - Finland

2- Projet AxIS, INRIA, Domaine de Voluceau, Rocquencourt, B.P. 105
78153 Le Chesnay Cedex - France

Abstract. Usually time series prediction is done with regularly sampled data. In practice, however, the data available may be irregularly sampled. In this case the conventional prediction methods cannot be used. One solution is to use Functional Data Analysis (FDA). In FDA an interpolating function is fitted to the data and the fitting coefficients are being analyzed instead of the original data points. In this paper, we propose a functional approach to time series prediction. Radial Basis Function Network (RBFN) is used for the interpolation. The interpolation parameters are optimized with a k-Nearest Neighbors (k-NN) model. Least Squares Support Vector Machine (LS-SVM) is used for the prediction.

1 Introduction

Time series prediction [1] is an important part of decision making in many application domains such as climatology and electricity network management. Usually linear models or neural network based methods are used for this task. Past values are used as inputs to the model and the output provides an estimate for the next value. These methods, however, have a serious limitation: the time series must be regularly sampled. In some application fields, such as medical time series, irregular sampling is frequent. Moreover, missing data are common in many real world application and correspond to a particular case of irregular sampling. In those cases conventional methods cannot be used.

One solution is to regenerate constantly sampled data by resampling the original data. This approach, however, is fairly noisy and thus not recommended. A better idea is to use Functional Data Analysis (FDA) [2]. In FDA the problem is casted into some function space where it can be analyzed more efficiently. This approach is based on the assumption that the data points are samples of some continuous function. The unknown function is estimated using some regression technique and the estimate is used as an input to the model. Neural networks that process functional data are called functional networks. They have been successfully applied to various data analysis tasks (see for example [6]).

In this paper, we introduce a functional time series prediction method inspired by functional auto-regressive model [3]. First a Radial Basis Function Network regressor is fitted to the data in time space. The fitting coefficients (i.e. weights of the RBF) are used as training examples for a Least Squares Support Vector Machine (LS-SVM) [4] prediction model.

The concept of functional networks is discussed in Section 2. In Section 3 the LS-SVM is presented briefly and in Section 4 the prediction method is outlined. An application to real world data is presented in Section 5.

2 Functional Networks

Functional networks stand for neural networks that process functional data instead of $\mathbb{R}^n$ data, using principals from Functional Data Analysis [2] For more detailed information about functional networks see for instance [6] and [8]. In literature functional data has been successfully applied to some neural networks models, mainly Radial Basis Function Networks (RBFN) and Multilayer Perceptrons (MLP) [6].

FDA can be considered as an extension to traditional data analysis. Multivariate analysis consists mainly of just vector operations, including norm and inner product. However, these operations are available on any Hilbert space and thus also on some function spaces such as L^2. Therefore much of the developed data analysis theory can be applied directly on functional data.

In practice functional data are never directly available: each function is known via a set of (input, output) pairs, possibly corrupted by noise. The first step is to try to estimate these underlying functions by projecting the data on some functional basis. Since function spaces in general are infinite dimensional, the basis must be truncated (as in the case of Fourier transform) or some approximation must be applied (as in the case of b-splines).

Let us assume that we have N observations and each observation consists of m_i pairs of measurements $(\mathbf{x}_j^i, y_j^i)_{j=1}^{m_i}$, where $\mathbf{x}_j^i \in \mathbb{R}^p$, $y_j^i \in \mathbb{R}$ and $i = 1, \ldots, N$. Basic assumption of FDA is that there is a regular function $f_i \in L^2$ such that $y_j^i = f_i(\mathbf{x}_j^i) + s_j^i$, where s_j^i stands for the observation noise. Knowing the truncated basis φ_i of the finite-dimensional function space $\mathcal{A}$ we can approximate f_i by minimizing the training error,

$$\min J(\mathbf{w}_i) = \sum_{j=1}^{m_i} \left(y_j^i - \hat{f}_i(\mathbf{x}_j^i) \right)^2 \text{ with } \hat{f}_i(\mathbf{x}) = \sum_{l=1}^{q} w_{i,l}\varphi_l(\mathbf{x}),$$

where $w_{i,l}$ are the fitting coefficients and q is the dimension of $\mathcal{A}$.

The function $\hat{f}_i$ is uniquely defined by the numerical regression coefficients $\mathbf{w}_i = [w_{i,1}, w_{i,2}, \ldots, w_{i,q}]^T$. Since the basis functions are not usually orthonormal (think to B-splines for instance) we transform the coefficients according to

$$\boldsymbol{\omega}_i = \mathbf{U}\mathbf{w}_i$$

where $\mathbf{U}$ is the Choleski decomposition $\mathbf{\Phi} = \mathbf{U}^T\mathbf{U}$ of the matrix $\mathbf{\Phi}_{i,j} = \langle \varphi_i, \varphi_j \rangle$. Euclidean operations (e.g., scalar product) on the transformed coefficients are equivalent to the corresponding operations in the functional space [6] (e.g. functional inner product): once the function estimates $\boldsymbol{\omega}_i$ have been obtained any conventional neural network model can be used for analysis.

3 LS-SVM for regression

LS-SVM is a least squares modification to the Support Vector Machine (SVM) [4]. The major advantage of LS-SVM is that it is computationally very cheap while it still possesses some important properties of the SVM. In this section we will briefly discuss the LS-SVM method for a regression task. For more detailed information see [4].

Again assume that we have a set of examples $(\boldsymbol{x}_i, y_i)_{i=1}^N$ and the goal is to estimate a function f as mentioned in the previous section. Basically we define a N dimensional function space by defining the mappings $\boldsymbol{\varphi} = [\varphi_1, \varphi_2, \ldots, \varphi_N]^T$ according to the measured points.

The LS-SVM model is of the form $\hat{f}(\boldsymbol{x}) = \mathbf{w}^T \varphi(\boldsymbol{x}) + b$ where $\mathbf{w}$ is a weight vector and b is a bias term. The optimization problem is the following,

$$\begin{aligned}\min J(\mathbf{w}, \boldsymbol{\epsilon}) &= \frac{1}{2}\mathbf{w}^T\mathbf{w} + \gamma\frac{1}{2}\sum_{i=1}^N \epsilon_i^2 \\ \text{so that} \quad y_i &= \mathbf{w}^T\boldsymbol{\varphi}(\boldsymbol{x}_i) + b + \epsilon_i, \quad i = 1, \ldots, N\end{aligned}$$

where the fitting error is denoted by ϵ_i. Hyper-parameter γ controls the trade-off between the smoothness of the function and the accuracy of the fitting. This optimization problem leads to a solution,

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^N \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i) + b$$

where α_i are the coefficients and $K(\mathbf{x}, \mathbf{x}_i) = \boldsymbol{\varphi}^T(\mathbf{x})\boldsymbol{\varphi}(\mathbf{x}_i)$ is the kernel. A common choice for the kernel is the Gaussian RBF,

$$K(\boldsymbol{x}, \boldsymbol{x}_i) = e^{-\dfrac{\|\boldsymbol{x} - \boldsymbol{x}_i\|^2}{2\sigma^2}}.$$

4 Time Series Prediction with Functional Network

4.1 Methodology

Consider a time series $\{t_i, y_i\}_{i=1}^N$ where the time stamps t_i belong to a closed interval $[a, b]$. First the data is divided into input windows I_h and output windows O_h, with $h = 1, \ldots, \lfloor (b - a - 2L)/\delta \rfloor + 1$:

$$\begin{aligned}I_h &= \{(t_i, y_i) \mid a + (h-1)\delta \qquad \leq t_i < a + (h-1)\delta + L\}, \\ O_h &= \{(t_i, y_i) \mid a + (h-1)\delta + L \leq t_i < a + (h-1)\delta + 2L\}.\end{aligned}$$

All windows have the same length L on the time axis. The shift between two sequential windows is δ. In the case of regularly sampled data the sampling interval is a natural choice for δ.

A outline if the prediction schema is shown in Figure 1. On each window, the time series is considered as a function and is modelled by a RBF network

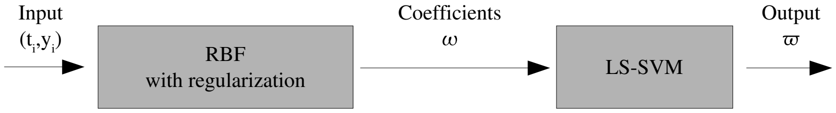

Fig. 1: Prediction method. First a RBF regressor is fitted to the data points. The prediction in done in the function space using LS-SVM. Thus the output is also a set of coefficients.

with Gaussian kernels. The centers of the kernels are equally distributed on an interval 10 per cent wider than the window. This ensures that the function $\hat{f}$ can be monotonically increasing at the borders of the window. In other words the model is

$$\tilde{f}(t) = \sum_{i=1}^{q} w_i K(t, t_i),$$

where t_i are the fixed centers and q is the number of kernels. Furthermore a regularization term $\boldsymbol{w}^T\boldsymbol{w}$ is used in the fitting to prevent overfitting [5].

The fitting coefficients are transformed using the Choleski decomposition as mentioned in Section 2. The obtained sets of input and output coefficients are denoted as $\mathcal{I}_h = \boldsymbol{\omega}(I_h)$ and $\mathcal{O}_h = \boldsymbol{\omega}(O_h)$ respectively.

Finally a prediction mapping $P : \mathcal{I}_h \mapsto \mathcal{O}_h$ in the function space is trained with LS-SVM. The kernel is also a Gaussian RBF. Since the dimension of the output is q, we are actually training q separate LS-SVM regressors for each dimension. The performance of the prediction is evaluated in a separate validation set. The mean square error at the known data locations is used as an error measure.

4.2 Optimizing parameters

There are basically four unknown parameters involved in the window function fitting: the window length L, number of kernels q, width of the kernels σ and the regularization parameter γ.

These parameters are optimized with a grid search. A predefined set of values is tested for each parameter. The parameter combination with the smallest Leave One Out (LOO) error is selected. Because we are exploring a four dimensional grid it is essential to speed up the testing process. For this purpose the fitting parameters are optimized using a k-NN approximator [7] [5] instead of LS-SVM. Since k-NN is computationally a very cheap method one is able to perform an larger search that would be feasible with LS-SVM. Of course the drawback is that the obtained parameters may not be optimal for the LS-SVM model.

Once the best fitting parameters are found a LS-SVM model is trained. The LS-SVM introduces two more unknown parameters $\tilde{\gamma}$ and $\tilde{\sigma}$. These parameters are also optimized with a grid search using LOO error.

5 Time Series prediction Application

5.1 Darwin dataset

The proposed prediction schema was tested with the Darwin dataset. It contains monthly values of sea level air pressures measured between years 1882 and 1998. An example of the data is shown in Figure 2 a). The first 1300 values of total 1400 were used for training and the remaining 100 values were used for validation.

Darwin dataset is constantly sampled. To experiment with missing data we randomly removed 33 per cent of the data points in the training set. In this experiment the mean square error was evaluated only in the first available data point.

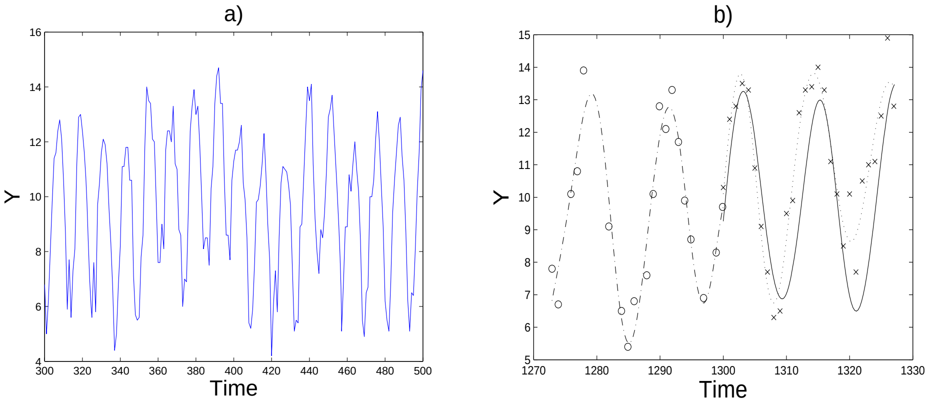

Fig. 2: a) Darwin dataset. b) Example of prediction.

5.2 Results

We performed several grid searches with k-NN model to tune the four fitting parameters. Figure 2 b) shows and example of the prediction with the best model (See Table 1). On the left there are the input data points and the input function while the correct output function and corresponding data points are on the right. The predicted output is marked with a solid line. A LS-SVM model was trained using these parameters. For reference the k-NN prediction was tested also without any data loss. The results are shown in Table 2.

L	q	σ	γ	k
27.1	8	1.70	348	25

Table 1: The best parameter setup for k-NN prediction with 33 % dataloss.

First of all it should be noted that the k-NN performance is very good when no data has been removed. Only the best conventional methods can reach better

	k-NN		LS-SVM
	0 % dataloss	33 % dataloss	33 % dataloss
LOO error	0.83	1.19	1.15
Test error	0.96	1.49	1.39

Table 2: Results of the Darwin dataset. This table contains mean square prediction errors on the learning set (LOO) as well as on the test set. k-NN was experimented with both 0 per cent and 33 per cent dataloss. Using the best setup of the latter case a LS-SVM model was trained.

mean square error. Naturally the error increases when one third of data is removed, but not so much as one would intuitively expect. Furthermore it can be seen that the LS-SVM model performs clearly better than k-NN.

6 Conclusions

We have proposed a functional LS-SVM time series prediction method to deal with the problem of irregular sampling or missing data. The method was experimented with the Darwin dataset. The results obtained with no data loss show that the functional approach is entirely comparable to the conventional methods. Furthermore the prediction performs reasonably well even if one third of the data points are missing.

In future, more tests should be done with irregularly sampled data.

References

[1] A. Weigend and N. Gershenfeld, editors. *Time Series Prediction: Forecasting the Future and Understanding the Past*, Addison-Wesley, Reading (MA), 1994.

[2] J. Ramsay and B. Silverman. *Functional Data Analysis*, Springer Series in Statistics, Springer Verlag, 1997.

[3] Besse, P., Cardot, H., Stephenson, D., 2000. Autoregressive forecasting of some functional climatic variations. Scandinavian Journal of Statistics 4, 673–688.

[4] J. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor and J. Vandewalle, *Least Squares Support Vector Machines*, World Scientific Publishing Co., Singapore, 2002.

[5] C. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Clarendon, 1995

[6] F. Rossi, N. Delannay, B. Conan-Guez, M. Verleysen, Representation of functional data in neural networks *Neurocomputing*, 64:183-210, Elsevier, 2005.

[7] A. Sorjamaa, N. Reyhani, A. Lendasse, Input and Structure Selection for k-NN Approximator. In J. Cabestany, A. Prieto, F. Sandoval, editors, proceedings of the 8^{th} *International Workshop on Artificial Neural Networks* (IWANN 2005), Lecture Notes in Computer Science 3512, pages 985-991, Springer-Verlag, 2005.

[8] B. Hammer and B. Jain, Neural Methods for non-standard data In M. Verleysen, editor, *proceedings of the* 12^{th} *European Symposium on Artificial Neural Networks* (ESANN 2004), d-side pub., pages 281-292, April 28-30, Bruges (Belgium), 2004.