A Bayesian Approach to Combined Neural Networks Forecasting

Maurits D. Out and Walter A. Kosters

Leiden Institute of Advanced Computer Science
Universiteit Leiden
P.O. Box 9512, 2300 RA Leiden, The Netherlands
Email: {mout,kosters}@liacs.nl

Abstract. Suitable neural networks may act as experts for time series predictions. The naive prediction is in a Bayesian manner used as prior to steer the weighted combination of these experts.

The paper was presented at ESANN'2000, The 8th European Symposium on Artificial Neural Networks, in Brugge (Belgium), April 26–28, 2000.

1. Introduction

Predicting the near future using information from the past is a challenging task. In this paper we try to do so by combining neural networks and techniques from Bayesian statistics (cf. [1] and [6]) in order to generate a prediction for an observable y_{t+1}, given the time series $y_1, y_2, \ldots, y_t$.

We start from the following observations. First it is hard to beat naive predictions—y_t in the setting above. Secondly, it is a complicated matter to combine expert forecasts into better ones. We feel that a joined effort might generate better results, especially if neural networks support the process. For this purpose we use so-called FIR networks (see [4]), where the usage of previous activations during the training stage makes them suited for time series analysis.

In this paper we use a Bayesian model for time series analysis. The prior is based upon the naive prediction. We then develop neural network techniques to provide the necessary expert results. These ingredients are combined into a joined forecast. Here we extend methods like those in [2].

Experiments show the potential of this approach. We conclude with suggestions for further research. Among other things we discuss different possibilities for the expert forecasts, all of them based upon neural network techniques.

2. Construction of a Bayesian Model

In this paper we consider a process $\{y_t : t = 1, 2, \ldots\}$ where y_t is a one-dimensional continuous random variable at time t. Suppose we are at time t

and we are given the task of predicting y_{t+1} given the sequence $y_1, y_2, \ldots, y_t$. We consider the following prior probability distribution of y_{t+1}:

$$(y_{t+1} \,|\, y_t) \sim N(y_t, \sigma^2). \tag{1}$$

This means that we assume that y_{t+1} has a Gaussian distribution with mean y_t and variance σ^2. Suppose that at the same time we have N expert forecasts for y_{t+1} available, contained in the column vector (T denoting transpose)

$$\vec{o}_t = (o_t^1, \ldots, o_t^N)^T. \tag{2}$$

We assume that each forecast o_t^i $(i = 1, 2, \ldots, N)$ is an unbiased estimator of y_{t+1}. This means that we can describe the relationship between $\vec{o}_t$ and y_{t+1} by

$$\vec{o}_t = y_{t+1} \cdot \vec{e} + \vec{v}, \tag{3}$$

where $\vec{e} = (1, 1, \ldots, 1)^T$ is the vector of dimension N consisting of ones and $\vec{v} = (v^1, v^2, \ldots, v^N)^T$ is the vector of random errors assumed to have a multivariate Gaussian distribution with mean zero (the N-dimensional zero vector) and a $N \times N$ variance-covariance matrix Σ. This means that the distribution of $\vec{o}_t$ given y_{t+1} (referred to as the likelihood of $\vec{o}_t$) can be described by

$$(\vec{o}_t \,|\, y_{t+1}) \sim N(y_{t+1} \cdot \vec{e}, \Sigma). \tag{4}$$

Starting from these observations we can now use Bayes' rule to derive the probability distribution of y_{t+1} given $\vec{o}_t$ and y_t, which is called the posterior distribution of y_{t+1}:

$$p(y_{t+1} \,|\, \vec{o}_t, y_t) = \frac{p(y_{t+1} \,|\, y_t) \cdot p(\vec{o}_t \,|\, y_{t+1}, y_t)}{p(\vec{o}_t \,|\, y_t)}, \tag{5}$$

where $p(\cdot)$ denotes the probability density function. In [1] it is shown that

$$(y_{t+1} \,|\, \vec{o}_t, y_t) \sim N(\mu_t, \xi^2), \tag{6}$$

where

$$\mu_t = W y_t + (1 - W)\, m_t, \quad \xi^2 = W \sigma^2, \tag{7}$$

$$W = 1 \,/\, \{\sigma^2 \cdot \vec{e}^T \Sigma^{-1} \vec{e} + 1\}, \tag{8}$$

$$m_t = \sum_{i=1}^{N} w^i \cdot o_t^i, \quad (w^1, w^2, \ldots, w^N) = \vec{e}^T \Sigma^{-1} /\ \vec{e}^T \Sigma^{-1} \vec{e}, \tag{9}$$

The mean μ_t in (7) is called the *Bayesian estimator*. It is easy to see that it is unbiased (i.e., the expectation of μ_t equals y_{t+1}). Another well-known result is that it is optimal in the sense that it is an estimator with minimal variance. We will use the Bayesian estimator as our forecast for the unknown y_{t+1}.

Note that the combination is carried out at two levels. First a weighted average of the expert forecasts based on the variance-covariance matrix is computed in (9). This results in the estimator m_t which is then combined with

y_t (the naive prediction) in (7) using both the variance-covariance matrix Σ and the variance σ^2 of the prior distribution in order to get the mean of the posterior distribution.

A problem however lies in the fact that in most situations Σ and σ^2 are unknown. One way to overcome this problem is to estimate these quantities using the known observations $y_1, y_2, \ldots, y_t$. We can for example estimate σ^2 by maximizing the likelihood L:

$$L(\sigma^2) = \prod_{k=2}^{t} p(y_k | y_{k-1}), \tag{10}$$

which gives us the following estimator:

$$\widehat{\sigma^2} = \frac{1}{t-1} \sum_{k=2}^{t} (y_k - y_{k-1})^2. \tag{11}$$

The variance-covariance estimator can also be estimated from $y_1, y_2, \ldots, y_t$ which yields the $N \times N$ matrix $\hat{\Sigma}$ with

$$(\hat{\Sigma})_{ij} = \frac{1}{t-2} \sum_{k=2}^{t} (v_k^i - v_{avg}^i)(v_k^j - v_{avg}^j) \quad (i, j = 1, 2, \ldots, N), \tag{12}$$

where $v_k^i = o_k^i - y_{t+1}$ and $v_{avg}^i = \frac{1}{t-1} \sum_{k=2}^{t} v_k^i$ denote the error of forecast i at time k and its average error, respectively.

In [2] several techniques that combine neural network forecasts are discussed. We mention *bumping*, i.e., take the network with the best performance, and *bagging*, i.e., take the unweighted mean of the networks.

3. Neural Networks as Expert Forecasts

In the previous section we discussed a Bayesian framework in which several expert forecasts were used to derive the posterior probability distribution of y_{t+1}. In our model we let feed-forward networks provide such forecasts. Their ability to discover non-linear relationships between an input space and a corresponding target space by means of examples makes them suitable models to generate predictions in time series. The main idea is to let each network generate a prediction of y_{t+1} when it is presented with an input pattern containing the sequence $(y_{t-M}, y_{t-M+1}, \ldots, y_t)$ for certain $M \geq 0$. Such input patterns are also referred to as time windows.

We choose standard feed-forward networks containing a single hidden layer, where the units have sigmoid activation functions. This layer is fully connected to the single neuron in the output layer which has the identity as its activation function. The output of this neuron will be the actual prediction of the network. As mentioned before we let the input consist of a time window containing the current observation and the M previous observations. This means that the output of the network is a function from the observation at the current time and from past observations. This construction may be viewed as a special instance of a FIR network (see [4]).

In the literature several strategies have been proposed for training and testing a feed-forward network for time series prediction. For our experiments we adopt the following scheme. Suppose we are at time t and we want to generate a prediction of the unknown y_{t+1}. We then compose a training set consisting of the S preceding time windows with corresponding targets. We train the network during a number of cycles on this training set by means of teacher-forcing adaptation, i.e., we do not feed the actual output of the network back as input, but take the real targets instead. We use the standard back-propagation algorithm to minimize the sum of squares error of the network on the training set:

$$E_t = \frac{1}{2} \sum_{s=1}^{S} (o_{t-s+1} - y_{t-s+1})^2, \tag{13}$$

where o_{t-s+1} denotes the output of the network when presented with the time window $(y_{t-s-M}, y_{t-s-M+1}, \ldots, y_{t-s})$.

4. Experiments and Results

In this section we present and discuss the results of experiments performed for two datasets. Three methods of constructing an ensemble (bumping, bagging and the Bayesian model described in Section 2.) and the naive prediction were implemented. For both datasets we constructed $N = 30$ feed-forward networks each having one hidden layer consisting of 5 units. At each point of time they were trained for 50 cycles on the $S = 20$ preceding time windows with $M = 10$. After training we let each technique make a prediction of the following element in the time series of which the error was computed. For each dataset the experiment was repeated 20 times and afterwards the average and the standard deviation of the root-mean-square error were computed.

The following two datasets were used:

The Mackey-Glass dataset: This is a dataset consisting of the first 1000 points of the series generated by the Mackey-Glass delay-differential equation with delay parameter 30, as described in [3].

Sunspots dataset: This dataset contains 280 yearly sunspot numbers, as described in [5].

In Table 1 the average root-mean-square errors yielded by the different techniques over 20 runs are presented. For bumping, bagging and the Bayesian model the standard deviation is also included. Note that the naive prediction has the same performance over the runs so its standard deviation was omitted.

For both datasets we see that the Bayesian model provides us with the lowest error. The improvement is significant when compared to the other techniques. To illustrate the performance of the Bayesian model, the time series

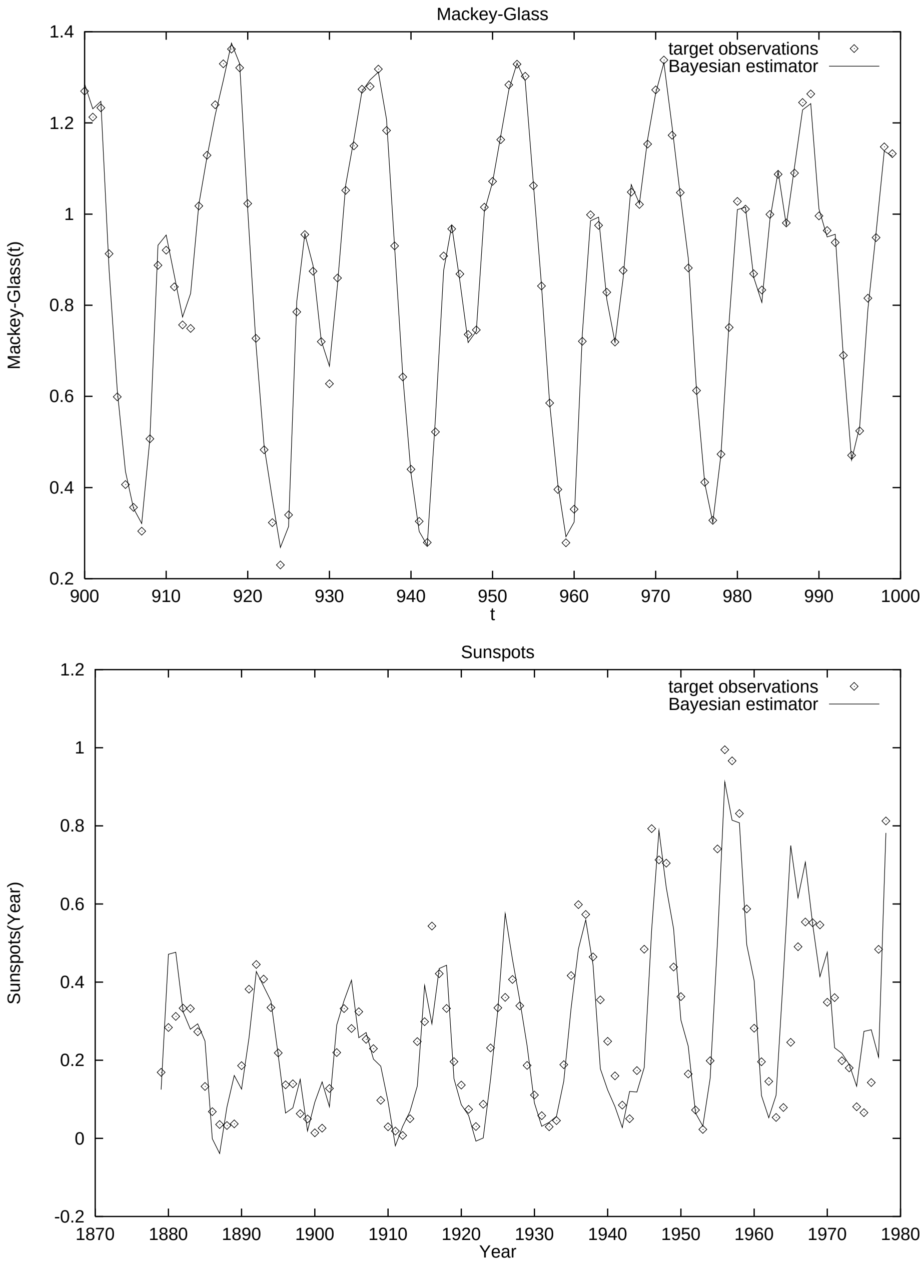

Figure 1: Graphs showing the performance of the Bayesian estimator on the Mackey-Glass series and the Sunspots series.

	Mackey-Glass		**Sunspots**	
	average	standard deviation	average	standard deviation
bumping	0.047	0.00056	0.397	0.03192
bagging	0.052	0.00059	0.335	0.00638
naive	0.380	-	0.382	-
Bayes	0.038	0.00077	0.323	0.00623

Table 1: The average and standard deviation of the root-mean-square errors over 20 runs for the two datasets.

and the function computed by Bayes' estimator for the last 100 observations of both datasets are shown in Figure 1.

5. Conclusions and Further Research

We may conclude that the approach provides promising results. The Bayesian combination of naive prediction and neural network experts is fruitful, and adheres to the intuition concerning expert forecasting.

For further research we are interested in the following. Contrary to other approaches the expert involved are neural networks. During the training stage it might be helpful to anticipate the later use of the outputs, and to ensure different behaviour for the networks. For instance, they might be trained into different directions in order to generate independent predictions. Variations on the methods from [2] need further study. We are also examining other underlying probability distributions, such as the generalized Laplace distribution.

References

[1] G. Anandalingam and L. Chen, *Linear Combination of Forecasts: A General Bayesian Model*, Journal of Forecasting **8** (1989), 199–214.

[2] T. Heskes, *Balancing Between Bagging and Bumping*, Advances in Neural Information Processing Systems **9** (1997), 466–472.

[3] M.C. Mackey and L. Glass, *Oscillation and Chaos in Physiological Control Systems*, Science **197** (1977), 287.

[4] E.A. Wan, *Time Series Prediction Using a Connectionist Network with Internal Delay Lines*, in A.S. Weigend and N.A. Gershenfeld (editors), Time Series Prediction: Forecasting the Future and Understanding the Past, SFI Studies in the Sciences of Complexity, Addison-Wesley, 1994.

[5] A.S. Weigend, B.A. Huberman and D.E. Rumelhart, *Predicting Sunspots and Exchange Rates with Connectionist Networks*, pp. 395–432 in M. Cas-

dagli and S. Eubank (editors), Nonlinear Modeling and Forecasting, Addison-Wesley, 1992.

[6] M.C. van Wezel, W.A. Kosters and J.N. Kok, *Maximum Likelihood Weights for a Linear Ensemble of Regression Neural Networks*, Proceedings ICONIP 1998, Japan, 498–501.