Comparing Gaussian Processes and Artificial Neural Networks for Forecasting

Tzai-Der Wang, Shang-Jen Chuang and Colin Fyfe
Applied Computational Intelligence Research Unit,
The University of Paisley,
Scotland. email:colin.fyfe@paisley.ac.uk

Abstract

We compare the use of artificial neural networks and Gaussian processes for forecasting. We show that Artificial Neural Networks have the advantage of being utilisable with greater volumes of data but Gaussian processes can more easily be utilised to deal with non-stationarity.

1 Introduction

Artificial neural networks [?] perform general purpose non-parametric regression and are modelled on an abstraction of real neural networks. They are supervised learning machines i.e. they require a set of training data on which an answer has previously been given. With respect to forecasting, we typically train on historical data and attempt to predict the future using what has been learned about the past.

A stochastic process $Y(\mathbf{x})$ is a collection of random variables indexed by $\mathbf{x} \in X$ such that values at any finite subset of X form a consistent distribution. A Gaussian Process (GP) therefore is a stochastic process on a function space which is totally specified by its mean and covariance function [?, ?, ?].

In this paper, we investigate the use of these two technologies in the context of forecasting airline passenger growth. Since neural networks are well known in the literature, we do not review them but do provide a short introduction to Gaussian Processes in the next section.

2 Gaussian Processes

Consider a stochastic process which defines a distribution, $P(f)$, over functions, f, where f maps some input space, χ to $\Re$. If e.g. $\chi = \Re$, f is infinite dimensional but the $\mathbf{x}$ values index the function, $f(\mathbf{x})$, at a countable number of points and so we use the data at these points to determine $P(f)$ in function space. If $P(f)$ is multivariate Gaussian for every finite subset of X, the process is a GP and is then determined by a mean function $\theta(\mathbf{x})$ and covariance function $\Sigma(\mathbf{x})$. These are often defined by hyperparameters, expressing our prior beliefs on the nature of θ and Σ, whose values are learned from the data.

A commonly used covariance function is $\Sigma : \Sigma_{ij} = \sigma_y^2 \exp(-\frac{(\mathbf{x}_i - \mathbf{x}_j)^2}{2l^2}) + \sigma_n^2 \delta_{ij}$ which enforces smoothing via the l parameter. The σ_y parameter determines the magnitude of the covariances and σ_n enables the model to explain the data, $y = f(\mathbf{x}) + n$, with $n \sim N(0, \sigma_n^2)$. Most commonly, we assume zero mean Gaussian processes, $\theta(\mathbf{x}) = 0$. We use the training data to estimate the parameters of the model. Let γ be a generic parameter of the covariance matrix, Σ. Then we use the standard method of gradient descent on the log likelihood with $t(\mathbf{x})$ as the target for training,

$$\frac{\partial L}{\partial \gamma} = -0.5 trace(\Sigma^{-1}\frac{\partial \Sigma}{\partial \gamma}) + 0.5 t(\mathbf{x})^T \Sigma^{-1} \frac{\partial \Sigma}{\partial \gamma} \Sigma^{-1} t(\mathbf{x})$$

where

$$\frac{\partial \Sigma}{\partial l} = 2\Sigma \frac{T}{2l^2}; \; \frac{\partial \Sigma}{\partial \sigma_y} = 2\sigma_y \exp(-\frac{T}{2l^2}); \; \frac{\partial \Sigma}{\partial \sigma_n} = 2\sigma_n I_N$$

where $T_{kj} = \frac{\|\mathbf{x}_k - \mathbf{x}_j\|^2}{2l^2}$.

Then we use the trained GP to predict new values, $f_* = f(\mathbf{x}_*)$ for test inputs, $\mathbf{x}_*$ using the fact that the combined distribution of all values is jointly Gaussian with

$$\left[\begin{array}{c} \mathbf{y} \\ \mathbf{f}_* \end{array} \right] \sim N\left(0, \left[\begin{array}{cc} K(X,X) + \sigma_n^2 I & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{array} \right]\right). \tag{1}$$

which leads to the predictive equations

$$\mathbf{f}_* | X, \mathbf{y}, X_* \quad \sim \quad N(\bar{\mathbf{f}}_*, \mathrm{cov}(\mathbf{f}_*)), \text{ where}$$

$$\begin{aligned} \bar{\mathbf{f}}_* &= E(\mathbf{f}_* | X, \mathbf{y}, X_*) \\ &= K(X_*,X)[K(X,X) + \sigma_n^2 I]^{-1}\mathbf{y} \qquad (2) \\ \mathrm{cov}(\mathbf{f}_*) &= K(X_*,X_*) \\ &\quad -K(X_*,X)[K(X,X) + \sigma_n^2 I]^{-1}K(X,X_*) \end{aligned}$$

3 Simulations

It is well-known [?] that the scalar product kernel

$$\Sigma : \Sigma_{ij} = \sigma_z^2 \mathbf{x}_i^T \mathbf{x}_j + \sigma_n^2 \delta_{ij} \tag{3}$$

can model non-stationary data. However this linear kernel only gives a linear response, but if the covariance matrix includes a trend aspect and also a local aspect which can create non-parametric regressions, we my model the type of data in which we are interested, as shown below. This is possible since the sum of two kernels (in this case the squared exponential and the linear) is itself a valid kernel.

Therefore to predict data such as that of Figure ??, we use a covariance matrix of the form

$$\Sigma : \Sigma_{ij} = \sigma_y^2 \exp(-\frac{(\mathbf{x}_i - \mathbf{x}_j)^2}{2l^2}) + \sigma_z^2 \mathbf{x}_i^T \mathbf{x}_j + \sigma_n^2 \delta_{ij} \tag{4}$$

and have the parameters learned in order to best model the data. This means that we have to find the optimal σ_z parameter for which we use

$$\frac{\partial \Sigma}{\partial \sigma_z} = 2\sigma_z T_z \tag{5}$$

Where $(T_z)_{ij} = \mathbf{x}_i^T \mathbf{x}_j$.

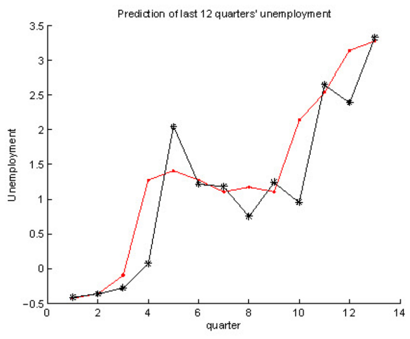

Figure 1: The actual and predicted unemployment on the test samples.

3.1 Unemployment Data

From [?], we have a set of 60 quarters' unemployment data from the USA which exhibits seasonal as well as non-stationary aspects. We use this data set to illustrate an investigation into the use of different covariance functions.

We have found it necessary to have the σ_n term learn much more slowly than the other terms, otherwise the data is totally explained by the noise. However, when we do have a lower rate for the noise we find that the σ_z term (which corresponds to the scalar product part of the covariance matrix) takes a value of 0.9, σ_n takes a value of 0.3 but σ_y takes a value of 0: the data is best explained by a trend term (see below) and the noise term. Predictions from this model on the last 12 quarters (not seen in training) are given in Figure ??. However when we train a GP with a covariance matrix only having a squared exponential term, we find it equally well explains the data but there is now a substantial noise term.

3.2 Monthly Airline Passengers

The monthly total number of UK air passengers from 1949 to 1999 (612 samples) is shown in Figure ?? (this data set is also taken from [?]). The seasonal cycle is clear as is the upward trend in the number of passengers.

We create 12 dimensional inputs from 12 consecutive samples of the univariate airline data and at-

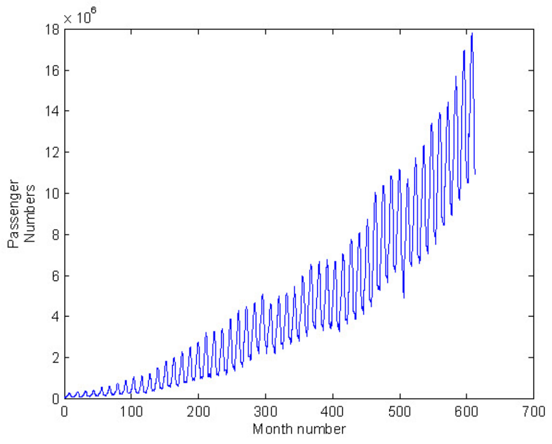

Figure 2: The total number of air passengers in the UK each month from 1949 to 1999.

tempt to predict the 13th sample. We use the technologies on the first 100 samples - 63 for training and 25 for testing; we will discuss why we use such a small sample later. We firstly use a multilayered perceptron (mlp) with the same data without using any pre-processing i.e. in particular not de-trending the data. In Figure ??, we show the results of two simulations the first with 20 hidden neurons, the second with 100 (which gave our best result with this data set[1]). In each case, we trained for 300000 iterations with a learning rate which decreased arithmetically from 0.01 to 0 during the course of the simulation. We see that the general shape of the prediction is correct but there is a distinct bias - the mlp is unable to respond automatically to the trend in the data and so we would have to use a pre-processing step which removes the trend. The MAPE for the latter experiment was 9.9%.

For the Gaussian Processes, the standard covariance matrix given in Section 2 does not work very well with this data: it captures the periodic nature of the data but is unable to respond to the upward trend and so gives cycles of the correct period but only with the average amplitude. Therefore we use the covariance function which can handle the upward

[1]We experimented with the number of hidden neurons varying between 2 and 200

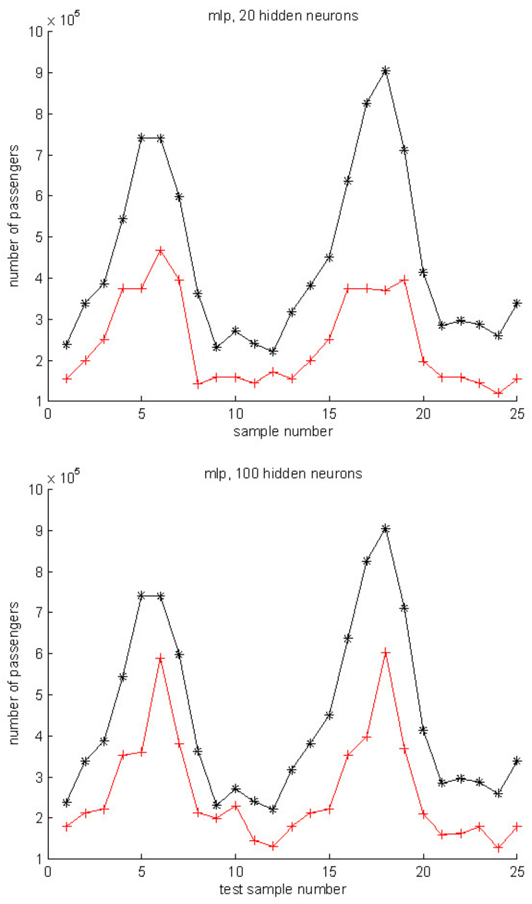

Figure 3: The multilayered perceptron is unable to identify the trend term in the data. Left: 20 hidden neurons. Right:100 hidden neurons.

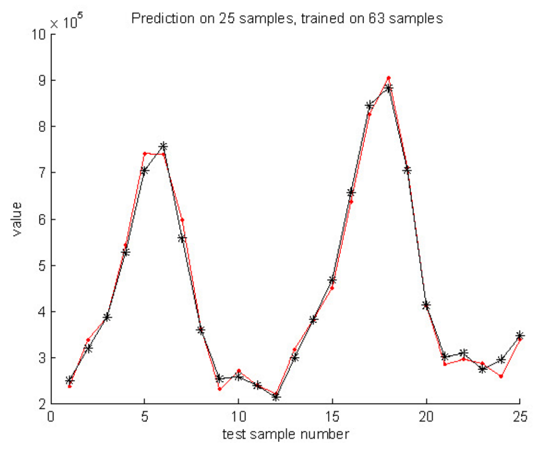

Figure 4: The red ”.”s are the actual values of the test data, the black ”*”s are the predicted values.

trend in the data and get the result shown in Figure ??. The MAPE is 3.83 on this sample. However there is a problem with this type of data set with the 575 samples: the covariance matrix becomes very ill-conditioned due to the scalar product influence i.e. the very feature which enables it to respond to the trend acts against accurate results on the whole data set.

It is worth recording that the trend aspect of this seems to be more difficult to model than the periodic. For example, [?] note an example from MacKay in which the one dimensional input variable x is decomposed into a two dimensional variable $\mathbf{u}(x) = (\cos(x), \sin(x))$ and the squared exponential function gives

$$\begin{aligned} K(x_1, x_2) &= \exp(-\frac{(\cos x_1 - \cos x_2)^2 + (\sin x_1 - \sin x_2)^2}{l^2}) \\ &= \exp(-\frac{2\sin^2(\frac{x_1 - x_2}{2})}{l^2}) \end{aligned}$$

We have had little success modelling this data set with this function.

4 Discussion

We have shown how, with a covariance function of the appropriate form, a Gaussian process can be used to model a non-stationary time series but one with a cyclical component. The artificial neural network trained using the backpropagation algorithm was not able to model this data. However, we used only a small subset of the training data. This is because the computational requirements for Gaussian processes are much greater than those for artificial neural networks: the neural network could easily be trained on all the data at once. The Gaussian process has another important feature - it can handle missing data more readily than the neural network. We do not believe that Gaussian processes will supplant neural networks (or any other non-parametric regressors) but see them as an additional tool in the data analyst's kit bag.

References

[1] C. Chatfield. *Time-series Forecasting.* Chapman and Hall, 2001.

[2] S. Haykin. *Neural Networks- A Comprehensive Foundation.* Macmillan, 1994.

[3] D. J. C. MacKay. Introduction to gaussian processes. Technical report, University of Cambridge, http://www.inference.phy.cam.uk/mackay/gpB.pdf, 1997.

[4] C. E. Rasmussen. *Advanced Lectures on Machine Learning*, chapter Gaussian Processes in Machine Learning, pages 63–71. 2003.

[5] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning.* MIT Press, 2006.

[6] C. K. I. Williams. Prediction with gaussian processes: from linear regression to linear prediction and beyond. Technical report, Aston University, 1997.