GENERALIZATION FOR MULTILAYER NEURAL NETWORK BAYESIAN REGULARIZATION OR EARLY STOPPING

CHI DUNG DOAN
SHIE-YUI LIONG

Department of Civil Engineering, National University of Singapore, Singapore - 119260

The study presents two approaches to increase the generalization capability, or to overcome the over-fitting tendency, of neural networks so that their prediction accuracies for unseen data can be further enhanced. The use of early stopping and Bayesian regularization approaches are considered. Data used are the artificial Mackey-Glass time series and the real time series of Mississippi River. Results show that the Bayesian regularization approach is best to overcome the over-fitting problems. It is observed that in the scenario when the data set considered is quite clean and large in size, the over-fitting effect is very small; thus, only marginal prediction improvement can be expected from the proposed approaches.

INTRODUCTION

Over-fitting problem or poor generalization capability happens when a neural network over learns during the training period. As a result, such a too well-trained model may not perform well on unseen data set due to its lack of generalization capability. Several approaches have been suggested in literature to overcome this problem. The first method is an early learning stopping mechanism in which the training process is concluded as soon as the overtraining signal appears. The signal can be observed when the prediction accuracy of the trained network applied to a test set, at that stage of training period, gets worsened. The second approach is the Bayesian Regularization. This approach minimizes the over-fitting problem by taking into account the goodness-of-fit as well as the network architecture.

Both approaches are considered in this study and demonstrated on (1) an artificial and a real time series data; (2) data of various noise-levels and sizes.

A very brief introduction of multilayer perceptron neural network together with back propagation learning algorithm is first given. This is followed by the measures to overcome poor generalization and the data used. Results of networks with or without the use of the approaches are compared and conclusions are finally drawn.

MULTI-LAYER PERCEPTRONS (MLP)

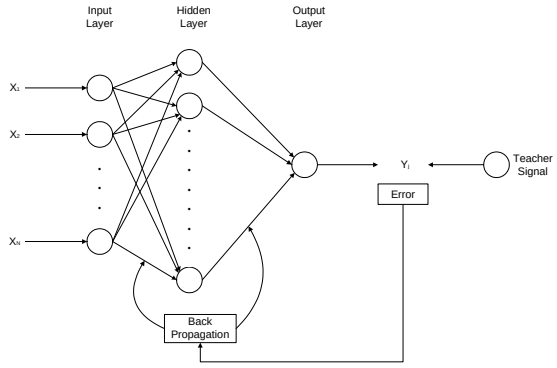
Artificial Neural Network (ANN) is a computing paradigm designed to mimic the human brain and nervous systems, which are primarily made up of neurons. The typical neural network is Multilayer Perceptrons (MLP). This type of network consists of the input, the

hidden and the output layers of neurons. Training MLPs in a supervised manner with the error back-propagation algorithm, many studies have shown MLPs ability to solve complex and diverse problems.

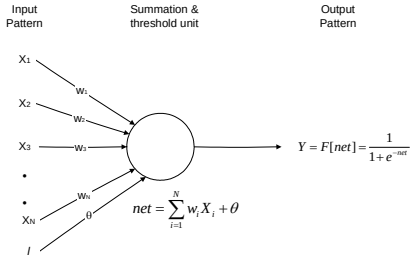
The error back-propagation learning consists of 2 passes through the different layers of the network: a forward and a backward passes. In the forward pass, an activity pattern is applied to the input neurons of the network and its effects propagate through the network, layer by layer, until an output is produced the network. Weights between neurons of successive layers are initially assigned in random. In the backward pass, the error observed between the network and the desired responses is computed and used to amend the weights. Each neuron starting from the hidden layer is modeled with a nonlinear activation function. The widely used function is the logistic expressed as:

$$y_j = \frac{1}{1+e^{-v_j}} \tag{1}$$

where y_j is the output of the neuron and v_j is the weighted sum of all inputs and the bias of neuron j. Figure 1 show the graphical composition of the neural network and the two computations done in each neurons.



(a) Composition of a Neural Network Model



(b) Two computational steps in each neuron

Figure 1. Schematic Configuration of a Neural Network

Learning by Scaled Conjugate Gradient (SCG) algorithm

In back propagation learning family, there are a number of algorithms available such as gradient descent, gradient descent with momentum, conjugate gradient, quasi-Newton. This study considers the Scaled Conjugate Gradient (SCG), developed by Moller (1993); it is based on a well known optimization technique in numerical analysis called the Conjugate Gradient Method. Unlike many other standard backward propagation algorithms, this technique does not require any user-specified parameters and its computation is faster and inexpensive. Detailed description of the algorithm can be found in [8].

APPROACHES TO AVOID OVER-FITTING PROBLEMS

Approaches considered overcoming the over-fitting problems are: (a) early stopping approach; (b) Bayesian Regularization approach.

Early Stopping Approach

This approach requires the data set to be divided into three subsets: training, test, and verification sets. The training and the verification sets are the norm in all model training processes. The test set is used to test the trend of the prediction accuracy of the model trained at some stages of the training process. At much later stages of training process, the prediction accuracy of the model may start worsening for the test set. This is the stage when the model should cease to be trained to overcome the over-fitting problem.

Bayesian Regularization Approach

The Bayesian Regularization approach involves modifying the usually used objective function, such as the mean sum of squared network errors (MSE or E_d)

$$F = E_d = \frac{1}{N}\sum_{i=1}^{N}(e_i)^2 \tag{2}$$

The modification aims to improve the model's generalization capability. The objective function in Eq. (2) is expanded with the addition of a term, E_w which is the sum of squares of the network weights:

$$F = \beta E_d + \alpha E_w \tag{3}$$

where the α and β are parameters which are to be optimized in Bayesian framework of MacKay ([3], [4]). It is assumed that the weights and biases of the network are random variables following Gaussian distributions and the parameters are related to the unknown variances associated with these distributions. It is a known fact that the optimal regularization technique requires quite costly computation of the Hessian matrix. To overcome this drawback, Gauss-Newton approximation to the Hessian matrix is used. The approximation with Levenberg-Marquardt algorithm for network training ([1], [2], [6], [9]) is used in this study.

DATA

Mackey-Glass Time Series

A noise-free artificial Mackey-Glass (MG) time series [5] is considered. The analysis is first applied on MG time series contaminated with various known noise levels measured in signal-to-noise-ratio (SNR) expressed as:

$$\text{SNR}=10\log_{10}\left(\frac{signal\ power}{noise\ power}\right) \tag{4}$$

Value of SNR can be interpreted as: smaller SNR signal implies that it has a higher noise level. A signal with a very large SNR is thus quite clean. If the noise level is defined as the ratio between variance of noise to the variance of signal, the relationship between the noise level and SNR is shown in Figure 2 and expressed as:

$$\text{noise\_level}=\frac{1}{10^{\text{SNR}/10}} \tag{5}$$

MG time series is written as:

$$\frac{dx(t)}{dt}=-0.1x(t)+\frac{0.2x(t-\Delta)}{1+x^{10}(t-\Delta)} \tag{6}$$

where Δ is the shift parameter. The equation can be solved numerically using the fourth-order Runge-Kutta method with the initial condition $x(t = 0) = 1.2$ and $x(t-\Delta) = 0.0$ for $0 \le t < \Delta$. A time step of 0.01 is used to solve the equation numerically. A moderately chaotic MG time series corresponding to $\Delta = 30$ is selected for study.

To analyze the performance of generalization approaches, MG time series data is injected with some known noise levels, the SNR values of 3, 10, 15, and 25. Different data sizes (300, 600, 3000) are also considered, Table 1.

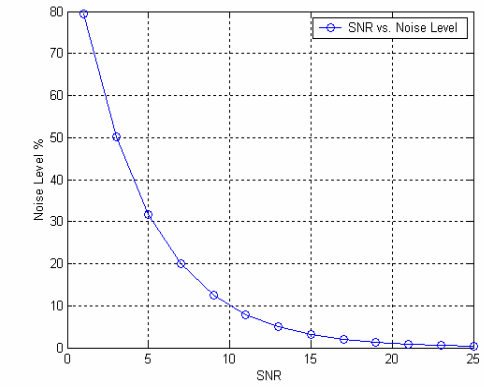


Figure 2. Relationship between noise level and SNR

Table 1. Various Data Sizes and SNR for Mackey Glass Time Series

Total Data Size	SNR	Equivalent Noise Level (%)	Training Size	Verification Size
300	3, 10, 15, 25	50.1, 10, 3.2, 0.3	200	100
600	3, 10, 15, 25	50.1, 10, 3.2, 0.3	400	200
3000	3, 10, 15, 25	50.1, 10, 3.2, 0.3	2000	1000

Mississippi River Time Series

A real time series, Mississippi River flow data, is also considered. Daily Mississippi river flow time series at Vicksburg station (1985-1993) with data sizes of 300, 600, and 3000 are used in the analysis. Data are obtained from the US Geological Survey website. The flow rate of the Mississippi river is quite large (mean at around 18,500m^3/s) as shown in Figure 3. The various sizes for training and verification follow exactly that of MG time series (Table 1).

A test set is required only when the early stopping criteria approach is considered. The test set data is taken from the training set as given in Table 1. Thus, the test set and the resulting training set together form the entire training data set given in Table 1.

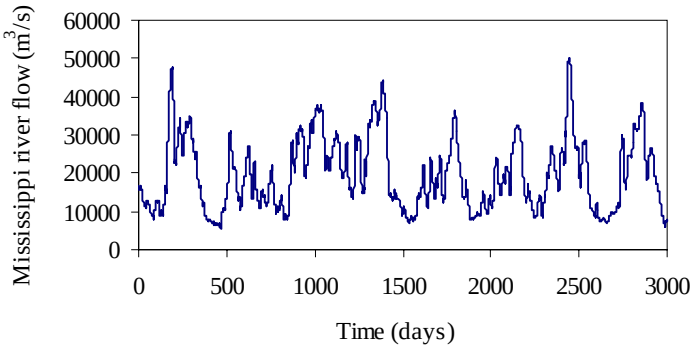


Figure 3. Mississippi River Flow Time Series

METHODOLOGY

The Multilayer Perceptrons is first constructed with 3 layers: input, hidden and output layers. The number of neurons in input layer is 6 representing the immediate past 6 data to forecast the next data in the time series. The number of neurons in hidden layer used, as suggested in [11], is

$$N = \text{integer}(n/2 + \sqrt{R}) \tag{7}$$

where n is the number of input neurons and R is the data set size.

This architecture of neural network is then trained by: (a) Scaled Conjugate Gradient (SCG); and (b) Bayesian Regularization (BR) with the configurations described in Table 2.

Table 2. Configurations of Running Neural Network

Configuration	Description
1	SCG trained without early stopping criteria
2	SCG with early stopping criteria; test set size : training set size = 1: 2
3	SCG with early stopping criteria; test : training = 1:3
4	Bayesian Regularization approach

RESULTS AND DISCUSSIONS

The results of Verification set for MG and Mississippi river time series, with various configurations, are presented in Tables 3 and 4 respectively. The highest prediction accuracy values are highlighted in grey and the second highest values are marked with bold italic font.

Table 3. Results of the Mackey Glass Time Series

Data Size	Noise (SNR)	Neural Network Configuration			
		1	2	3	4
300	3	-0.95507	***0.43335***	0.44990	0.38446
	10	0.55079	0.81674	***0.80814***	0.79938
	15	0.85700	0.81928	***0.92744***	0.92870
	25	***0.98637***	0.97166	0.97568	0.98979
600	3	-0.18162	***0.45672***	0.44188	0.45683
	10	0.76698	0.83425	***0.83140***	0.82733
	15	0.91980	0.93106	***0.93347***	0.93569
	25	***0.99130***	0.98010	0.98170	0.99137
3000	3	0.43427	0.46636	***0.46681***	0.47944
	10	***0.82499***	0.82247	0.82193	0.83429
	15	***0.93906***	0.92684	0.92612	0.93985
	25	***0.99259***	0.98898	0.98875	0.99265

Table 4. Results of the Mississippi River Time Series

Data Size	Neural Network Configuration			
	1	2	3	4
300	***0.99775***	0.95771	0.83777	0.99851
600	0.99838	0.98297	0.98695	***0.99800***
3000	0.99787	0.99713	0.99597	***0.99751***

From Table 3, the following main observations are made:

- For all cases when noise level is very high (SNR=3), it is clear that forecasting with early stopping criteria (Config. 2 and 3) or with BR (Config. 4) approach is better than without stopping criteria (Config. 1);
- For cases when the noise level is very low (SNR=25), ANN without early stopping criteria or BR approach performs almost equally well;
- Small data size combined with very noisy data should not train ANN without generalization. A poor performance can be expected otherwise.
- In general, the Bayesian Regularization approach performs better in most of the cases.

The Mississippi river time series (Table 4) agrees with the observations obtained from the MG time series with low noise levels. ANN without early stopping criteria or BR approach yields equally good performance.

CONCLUSIONS

The study examined the generalization ability of several approaches on neural network forecasting model. The approaches were first tested on artificial clean data which are contaminated with noises of known levels. A real time series data, the daily Mississippi river flow, was also considered in the study.

Results showed that, in general, the Bayesian Regularization (BR) approach, compared to the early stopping approach, lends the model higher generalization ability. Thus, BR yields higher prediction accuracy than the early stopping approach. Also another advantage of BR over early stopping approach is that, BR does not require a test set. It should be noted that the length of the test set has an impact on the prediction capability of the trained model.

The prediction improvement of models trained with generalization approaches over those with standard approach is considerably remarkable when the data is very noisy. However, this prediction improvement diminishes significantly when data set considered is quite clean. In other words, generalization approach does not play a crucial role in training neural network when data set used is of large size and relatively clean.

REFERENCES

[1] Foresee, F.D. and Hagan, M.T., Gauss-Newton Approximation to Bayesian Regularization. Proceedings of the 1997 International Joint Conference on Neural Networks, pp. 1930-1935, 1997.

[2] Levenberg, K., A Method for the Solution of Certain Problems in Least Squares. Quarterly Applied Mathematics 2, pp. 164-168, 1944.

[3] MacKay, D.J.C., A Practical Bayesian Framework for Backpropagation Networks. Neural Computation, Vol. 4(3), pp. 448-472, 1992.

[4] MacKay, D.J.C., Bayesian Interpolation. Neural Computation, Vol. 4(3), pp. 415-447, 1992.

[5] Mackey, M. and Glass, L., Oscillation and Chaos in Physiological Control Systems. Science, Vol. 197, pp. 287-289, 1977.

[6] Marquardt, D., An Algorithm for Least-squares Estimation of Nonlinear Parameters. SIAM Journal Applied Mathematics, Vol. 11, pp. 431-441, 1963.

[7] MATLAB User Manual. The MathWorks Inc., 2000.

[8] Moller, M. F., A scaled conjugate gradient algorithm for fast supervised learning. Neural Networks, Vol. 6, pp. 525-533, 1993.

[9] More, J.J., The Levenberg-Marquardt Algorithm: Implementation and Theory. Numerical Analysis, edited by. G. A. Watson, Lecture Notes in Mathematics 630, Springer Verlag, pp. 105-116, 1977.

[10] Neural Network Toolbox, User Guide. The MathWorks Inc., 2000.

[11] NeuroShell 2, Software by Ward Systems Group, Inc.