TRANSPARENT NEURAL NETWORK MODELS USING PRODUCT UNIT BASED NEURAL NETWORKS

D.S.K. Karunasingha[1] and A.W. Jayawardena[2]

[1]Department of Engineering Mathematics, Faculty of Engineering, University of Peradeniya
[2]International Centre for Water Hazard and Risk Management (ICHARM) under the auspices of UNESCO, Public Works Research Institute, Tsukuba, Japan

Introduction

Many successful applications of Multi-layer Perceptrons (MLP), also called Sigmoidal Neural Networks, have appeared in different problems in hydrology in the last 2 decades. However, one frequent criticism towards ordinary ANN models is their lack of transparency. Multiplicative neural networks have been proposed in the literature as an alternative. Product Unit based Neural Networks (PUNN) is a type of multiplicative neural networks, which has the advantages of increased information capacity and the ability to form higher order combinations of the inputs. However, the major drawback of this type of networks is the difficulty in training with the back-propagation algorithm compared with standard sigmoidal networks. Recently, Martinez-Estudillo et al. (2006) have proposed a model of evolutionary programming for automatically obtaining the structure and the weights of product units based neural networks. They have proved that the product units they have used have universal approximation capability. Karunasingha et al. (2010) have consolidated the proposed algorithm and have shown their ability to obtain transparent models compared to MLP through a hydrological application.

Our objective is to demonstrate the readability and the robustness of the models produced by PUNN, and to compare their performance with the two most competitive machine learning techniques, multi-layer perceptrons (MLP) with sigmoid neurons and additive units, and support vector machines (SVM) with Gaussian kernel. A river flow time series prediction problem is used for this purpose.

Product unit based neural networks

Martinez-Estudillo et al. (2006) considered three-layer networks where there is an input layer, a single hidden layer of product units and an output layer of additive units. The evolutionary algorithm of Martinez-Estudillo et al. (2006) to train this type of networks uses two mutation operators namely parametric mutation and structural mutation, and the elitism strategy. The parametric mutation updates the weights of PUNN using a simulated annealing algorithm (Kirkpatrik et al., 1983). The structural mutation modifies the structure of the network.

Methodology

We have implemented the above evolutionary algorithm for training PUNN, in MATLAB. To demonstrate the pros and cons of PUNN compared

to SVM and MLP, the algorithm is applied to daily flow prediction problem in Nan River of Thailand. Current and two time-lagged values of flow are used as inputs to predict the flow at the next time level. For each of training, testing and validation sets, two years of data are used. All the input data are normalized to the range 0.1 – 0.9 and the output data to the range 1 – 2. For MLP, networks with a single hidden layer of logistic sigmoid neurons are used. The best MLP is chosen from a series of trials by changing various parameters. For SVM, Gaussian kernel with ε - insensitive cost function is used. Optimal parameters for SVMs are found using a micro-Genetic Algorithm (Krishnakumar, 1989). The parameters and methods used for PUNN are same as in Karunasingha et al. (2010).

Results

The errors on validation set with each technique for the best models (in terms of error on test set) are shown in Table 1. The optimal parameters for SVM were: the trade-off parameter $C = 43.95$, the width of the Gaussian kernel $\sigma = 0.05373$ and the tolerance in cost function $\varepsilon = 0.00011$. The best model obtained by PUNN is given in Eq. 1 and another model (of the same level of accuracy) produced by the evolutionary algorithm with a different seed number for the random number generator is given in Eq. 2.

$$y_t = 0.8841 + 0.54615 x_{t-3}^{0.1797} x_{t-2}^{-0.3851} x_{t-1}^{0.881} \quad (1)$$

$$y_t = 0.8955 + 0.54071 x_{t-3}^{0.2846} x_{t-2}^{-0.5487} x_{t-1}^{0.9} \quad (2)$$

Discussion

Table 1 shows that the PUNN model has obtained the same level of accuracy as the MLP and SVM in prediction of Nan River flow series. The simplicity, in terms of easy readability of the model, is the most appealing feature of the PUNN model as shown in Eq. 1 compared to the black-box models produced by MLP and SVM. The positive and negative

Table 1. Prediction errors

Method	Prediction errors on Validation data			
	NSE	NRMSE	MAE m^3/s	RMSE m^3/s
MLP	0.9055	0.3074	24.61	54.48
SVM	0.8956	0.3232	21.66	57.28
PUNN	0.9046	0.3089	25.70	54.76

NSE – Nash-Sutcliffe coefficient of efficiency
NRMSE – Normalized root mean square error
MAE – Mean absolute error
RMSE – Root mean square error

exponents with the positive coefficient in Eq. 1 show that the output increases with increasing x_{t-1}, x_{t-3} and decreases with increasing x_{t-2}. This is consistent with the sample partial autocorrelation function (SPACF) of the flow series which have significant positive, negative and positive values at lags 1, 2, and 3 respectively.

Models in Eq. 1 and Eq. 2 have the same form: both have a constant term and a term containing the three input variables. It should be noted that this form has been arrived at from a wide range of different structures possible. In addition, the coefficients and the exponents in the two models have the same signs and similar values. This robustness is also an interesting feature which is lacking in MLP and SVM. For example, if one trains two MLPs starting with different initial weights keeping all the other parameters such as number of hidden neurons same, the two networks do not converge to the similar values/signs in terms of weights, and also they do not even converge to the same structure if pruning is to be exercised.

The use of PUNN with evolutionary algorithm requires less expertise from the user compared to MLP where both the knowledge and the experience are necessary to build reasonable models. A drawback of evolutionary PUNN model is that, in its current form, it takes considerably higher computational time compared to MLP and SVM. Application of PUNN in different types of real world problems is useful to further explore its advantages and disadvantages.

References

Karunasingha, D. S. K., Jayawardena, A. W. and Li, W.K. (2010). Evolutionary Product Unit Based Neural Networks for Hydrological Time Series Analysis. (Accepted for publication in J. of Hydroinformatics).

Kirkpatrik, S., Jr. Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by Simulated Annealing. Science, 220(4598): 671-680.

Krishnakumar, K. (1989). Micro-Genetic Algorithms for Stationary and Non-Stationary Function Optimization. SPIE Conference on Intelligent Control and Adaptive Systems, 1196, Philadelphia, PA.

Martinez-Estudillo, A., Martinez-Estudillo, F., Hervas-Martinez, C. and Garcia-Pedrajas, N. (2006). Evolutionary Product Unit based Neural Networks for Regression. Neural Networks, 19: 477-486.