Neurocomputing 81 (2012) 108–112

Contents lists available at SciVerse ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Letters

Daily maximum load forecasting of consecutive national holidays using OSELM-based multi-agents system with weighted average strategy

Keem Siah Yap [a,*], Hwa Jen Yap [b]

[a] College of Graduate Studies, Universiti Tenaga Nasional, Jalan IKRAM-UNITEN, 43009 Kajang, Selangor, Malaysia
[b] Department of Engineering Design and Manufacture, Faculty of Engineering, University of Malaya, 50603 Kuala Lumpur, Malaysia

ARTICLE INFO

Article history:
Received 29 July 2011
Received in revised form
17 December 2011
Accepted 17 December 2011
Communicated by G.-B. Huang
Available online 27 December 2011

Keywords:
Gradient descent
Load forecasting
Multi-Agent System
Online Sequential Extreme Learning Machine
Weighted average

ABSTRACT

In the previous research, a Multi-Agent System based on Online Sequential Extreme Learning Machine (OSELM) neural network and Bayesian Formalism (MAS-OSELM-BF) has been introduced for solving pattern classification problems. However this model is incapable of handling regression tasks. In this article, a new OSELM-based multi-agent system with weighted average strategy (MAS-OSELM-WA) is introduced for solving data regression tasks. A MAS-OSELM-WA consists of several individual OSELM (individual agent) and the final decision (parent agent). The outputs of the individual agents are sent to the parent agent for a final decision whereby the coefficients of parent agent are computed by a gradient descent method. The effectiveness of the MAS-OSELM-WA is evaluated by an electrical load forecasting problem in Malaysia for a month with consequent national holidays (i.e., during the month of *Hari Raya*—Malay New Year of Malaysia). The results demonstrated that the MAS-OSELM-WA is able to produce good performance as compared with the other approaches.

1. Introduction

Short-term power Load Forecasting is very important from the power systems grid operation point of view. It involves forecasting load demand in a short-term time frame. It may be half hourly prediction up to weekly prediction. Accurate forecasting is important to the utility company for ensuring reliability and stability of the grid to meet the load demand, hence better savings while maintaining the security of the grid. From the survey as reported by Hobbs et al. [1], the largest beneficiary of electrical load forecast by neural network accrued USD7600,000 per year with lowering of MAPE (mean absolute percentage error) by 1.5%.

Solving the electrical power load forecasting problems using neural networks has been one of the major researches in data regression and neural networks. Unlike other types of data regression, load forecasting uses historical outputs of a problem as the training inputs to the neural network. A load forecasting problem with one output and M inputs can be modeled as $x_k = f(x_{k-1}, x_{k-2}, \ldots, x_{k-M+1}, x_{k-M})$ where function $f(.)$ is the model and x_k is the output at time-k. Generally, solving a load forecasting problem using neural network would have the same solution approach as the other types of regression problem with the exception that the inputs and output for the neural network would consist of historical data.

There have been several efforts in implementing Neural Network in the load forecasting algorithm, e.g. Support Vector Machine (SVM) [2] and other neural network model [3]. The result of these efforts opened up several avenues in improving the load forecasting results. However, load forecasting problem is solved by historical data only without taking into account external factors such as weather and significant events (e.g., public holidays). The disadvantage of this method is that an error in the forecasted values will occur if there are future events available in the historical data. In order to improve the accuracy of the time series model, a solution will be to incorporate external information, e.g. the type of the day (Monday, Tuesday, etc.), that forces the load profiles to represent a complete cycle in a week, as well as information of a public holiday that falls on a weekday to indicate a lower power demand although it is supposed to be a working day.

As load forecasting can be considered as a time series prediction that is used to be considered as a nonstationary process. The current state of knowledge for the nonstationary processes is considerably poorer than that of the stationary processes [4]. In many applications, nonstationary signals are treated in the form of stationary owing to the ease of analysis [4]. Nonstationary

* Corresponding author. Tel.: +603 89287309; fax: +603 89212065.
E-mail addresses: yapkeem@uniten.edu.my (K.S. Yap), hjyap737@um.edu.my (H.J. Yap).

0925-2312/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2011.12.002

processes are undoubtedly more difficult to analyze and their diversity makes application of universal tools impossible. Here, solving the electrical load forecasting problem by neural networks is considered as a regression task of nonstationary time-series processes [5].

On the other hand, the Multilayer Perceptron (MLP) [6,7] and Radial Basis Function (RBF) neural network [6,8] are the most popular learning strategy of neural networks. However, learning of a standard MLP and RBF networks is required with an adjustment in the weight in every iteration after all training samples are presented to the network. Hence, learning of MLP and RBF usually is time consuming because the learning process may involve many iterations through the training samples.

Recently, a neural network known as the Online Sequential Extreme Learning Machine (OSELM) is proposed [9]. It is a multilayer neural network capable of incremental learning network without having to add more hidden neurons. This is suitable to be used for handling nonstationary problems that information may be changed by time. On top of that, OSELM does not require adding more hidden neuron along the learning of solving nonstationary problems.

Several improved version of OSELM have been proposed, e.g., the enhanced-random-search OSLM [10] that is able to improve the performance of OSELM by pre-selection of hidden neurons, hybridization of fuzzy logic with OSELM [11], the error minimized OSELM that can grown hidden neurons one by one or group by group [12], ensemble OSELM [13] that combined outputs of all the multiple OSELMs by the average value and a OSELM-based multi-agent systems with Bayesian Formalism (MAS-OSELM-BF) for solving classification task [14].

In the most recent publication, Huang et al. [15] have highlighted the relationship and the advantages of ELM as compared with the least square support vector machine (LS-SVM) and proximal support vector machine (PSVM) for handling both regression and multi-class classification tasks. In addition, a detail survey of ELM-based neural network and applications can be found in [16]. Note that it is important to point out that the ELM-based neural networks are capable of universal approximation with random hidden nodes. This has been discussed and justified in [17,18,19].

Similar to MAS-OSELM-BF [14] for classification problems, the objective of this article is to present a new design of OSELM-based multi-agent systems but for regression task. In general, this can be done by replacing the Bayesian Formalism with a weighted average (WA) strategy and form a MAS-OSELM-WA. The effectiveness of the proposed MAS-OSELM-WA is tested with a nonstationary load forecasting problem, i.e., forecasting over a long consecutive national holidays.

This article is organized as follows. Section II describes the learning algorithms of OSELM and MAS-OSELM-WA. Section III presents the use of the proposed method for solving load forecasting of a long consecutive national holidays in Malaysia, i.e., the *Hari Raya* (Malay New Year of Malaysia). A summary of this paper, with suggestions for further work, is presented in Section IV.

2. The algorithms of OSELM and MAS-OSELM-WA

This section presents the review of OSELM and the algorithm of MAS-OSELM-WA.

2.1. OSELM

As shown in Fig. 1, OSELM is considered a feedforward or an RBF (depends to type of activation function, as following) with advanced learning algorithm. Consider a set of N training samples (with a input vector and a target output vector), $(\mathbf{x}_j,\mathbf{t}_j) \in \mathbf{R}^n \times \mathbf{R}^m$,

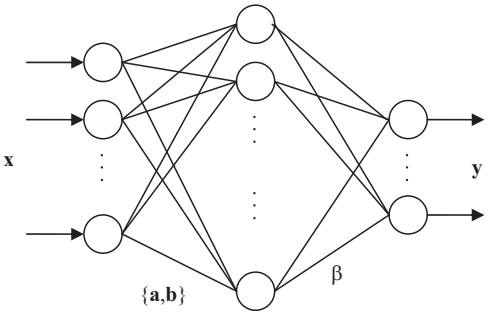

Fig. 1. Architecture of an OSELM.

are used for training an OSELM with L number of hidden nodes. In a perfect case, the output of this OSELM to $\mathbf{x}_j$ should be

$$f(\mathbf{x}_j) = \sum_{i=1}^{L} \beta_i G(\mathbf{a}_i,b_i,\mathbf{x}_j) = \mathbf{t}_j \text{ for } j = 1,\ldots,N \tag{1}$$

where $\mathbf{a}_i$ and b_i are the input weights and bias (learning parameters) of the hidden nodes, β_i is the output weight and $G(\mathbf{a}_i,b_i,\mathbf{x}_j)$ is the output of the ith hidden neuron to the input vector $\mathbf{x}_j$. Eqs. (2) and (3) show the definition of the $G(\mathbf{a}_i,b_i,\mathbf{x}_j)$ for additive hidden neuron and RBF neuron, respectively:

$$G(\mathbf{a}_i,b_i,\mathbf{x}_j) = \frac{1}{1+\exp\{-(\mathbf{a}_i\mathbf{x}_j+b_i)\}}, \quad b_i \in R \tag{2}$$

$$G(\mathbf{a}_i,b_i,\mathbf{x}_j) = \exp\{-b_i\|\mathbf{x}_j-\mathbf{a}_i\|^2\}, \quad b_i \in R^+ \tag{3}$$

There are two phases in the training of OSELM, i.e., the initialization phase and sequential learning phase. In the initialization phase, a small chunk of training data (denoted as N_0) is used to train the OSELM with L (where $N_0 \geq L$) number of hidden neurons. The procedures of the initialization phase are as follows:

– *initialization phase*

(a) Randomly assign the input weights $\mathbf{a}_i$ and b_i for $j = 1, \ldots, L$.

(b) Calculate the initial hidden layer output matrix, $\mathbf{H}_0$, as follows:

$$\mathbf{H}_0 = \begin{bmatrix} G(\mathbf{a}_1,b_1,\mathbf{x}_1) & \ldots & G(\mathbf{a}_L,b_L,\mathbf{x}_L) \\ \vdots & \ldots & \vdots \\ G(\mathbf{a}_1,b_1,\mathbf{x}_{N_0}) & \ldots & G(\mathbf{a}_L,b_L,\mathbf{x}_{N_0}) \end{bmatrix}_{N_0 \times L} \tag{4}$$

(c) Estimate the initial output weights, β^0, by the following equations:

$$\mathbf{P}_0 = (\mathbf{H}_0^T\mathbf{H}_0)^{-1} \tag{5a}$$

$$\beta^0 = \mathbf{P}_0\mathbf{H}_0\mathbf{T}_0 \tag{5b}$$

where $\mathbf{T}_0 = [\mathbf{t}_1,\ldots,\mathbf{t}_{N_0}]$ is the respective targeted output vectors.

(d) Set $k = N_0$ then go to sequential learning phase that enables learning other samples one-by-one, as the following sequential learning phase.

– *sequential learning phase*

(e) Present a new training samples, assuming it is the $(k+1)$th sample in the whole training samples.

(f) Compute the new hidden layer output matrix:

$$\mathbf{H}_{k+1} = \begin{bmatrix} G(\mathbf{a}_1,b_1,\mathbf{x}_k) & \ldots & G(\mathbf{a}_L,b_L,\mathbf{x}_{k+1}) \end{bmatrix} \tag{6}$$

(g) Calculate the output weights β^{k+1} by the following equations:

$$\mathbf{P}_{k+1} = \mathbf{P}_k - \mathbf{P}_k\mathbf{H}_{k+1}^T(\mathbf{I}+\mathbf{H}_{k+1}\mathbf{P}_k\mathbf{H}_{k+1}^T)^{-1}\mathbf{H}_{k+1}\mathbf{P}_k \tag{7a}$$