Neurocomputing 72 (2009) 3391–3395

Contents lists available at ScienceDirect

Neurocomputing

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

Letters

Ensemble of online sequential extreme learning machine

Yuan Lan *, Yeng Chai Soh, Guang-Bin Huang

School of Electrical and Electronic Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798, Singapore

ARTICLE INFO

Article history:
Received 3 November 2008
Received in revised form
20 February 2009
Accepted 21 February 2009
Communicated by T. Heskes
Available online 1 April 2009

Keywords:
Extreme learning machine
Ensemble
Online learning
Sequential learning

ABSTRACT

Liang et al. [A fast and accurate online sequential learning algorithm for feedforward networks, IEEE Transactions on Neural Networks 17 (6) (2006), 1411–1423] has proposed an online sequential learning algorithm called online sequential extreme learning machine (OS-ELM), which can learn the data one-by-one or chunk-by-chunk with fixed or varying chunk size. It has been shown [Liang et al., A fast and accurate online sequential learning algorithm for feedforward networks, IEEE Transactions on Neural Networks 17 (6) (2006) 1411–1423] that OS-ELM runs much faster and provides better generalization performance than other popular sequential learning algorithms. However, we find that the stability of OS-ELM can be further improved. In this paper, we propose an ensemble of online sequential extreme learning machine (EOS-ELM) based on OS-ELM. The results show that EOS-ELM is more stable and accurate than the original OS-ELM.

1. Introduction

Early in 1990, the idea of neural network ensembles has already been proposed by Hansen and Salamon [2]. Their work showed that a single network's performance can be expected to improve using an ensemble of neural networks with a plurality consensus scheme. This technique has been widely spread after that. The most prevailing approaches for training neural network ensembles are Bagging [3] and Boosting [4–6]. However, most of the methods used to train the ensemble require all the data available at the process of training, and it is usually referred to as batch learning. In many real world applications, learning has to be really an ongoing process since the complete set of data is usually not available at once. When new data arrive, batch learning performs a retraining using past data as well as the new data, and hence costs a lot of time.

Therefore, sequential learning algorithms could be a solution for networks learning new information as it becomes available. Online sequential extreme learning machine (OS-ELM) proposed by Liang et al. [1] is a fast and accurate online sequential learning algorithm for single hidden layer feedforward networks (SLFNs) with additive and radial basis function (RBF) hidden nodes. OS-ELM is developed on the basis of extreme learning machine (ELM) [7–12] that is used for batch learning and has been shown to be extremely fast with good generalization performance. Compared to ELM, OS-ELM can learn data one-by-one or chunk-by-chunk (a block of data) with fixed or varying chunk size. The parameters of hidden nodes in OS-ELM (input weights and biases for additive nodes or the centers and impact factors for RBF nodes) are randomly selected and the output weights are analytically determined. Simulation results in [1] have shown that OS-ELM is faster than other sequential algorithms and produces better generalization performances on many benchmark problems in the regression, classification and time-series prediction areas.

However, we find that similar to other learning algorithms OS-ELM may have variations in different trials of simulations. To improve the performance of OS-ELM and introduce the sequential learning mode into the ensemble networks, we propose an integrated network structure, which is called ensemble of online sequential extreme learning machine (EOS-ELM). EOS-ELM comprises several OS-ELM networks. The average value of outputs of each OS-ELM in the ensemble is used as the final measurement of network performance. The simulation results prove that EOS-ELM is more stable than original OS-ELM in each trial of simulation for most of problems. And we conduct the comparison between EOS-ELM and negative correlation learning (NCL) [13]. The results show that EOS-ELM is better than NCL on the applications selected.

2. Review of OS-ELM

OS-ELM on the basis of ELM was developed for SLFNs with additive and RBF hidden nodes. Consider N arbitrary distinct samples $(\mathbf{x}_i, \mathbf{t}_i) \in \mathbf{R}^n \times \mathbf{R}^m$. If a SLFN with L hidden nodes can approximate these N samples with zero error, it then implies that

* Corresponding author.
E-mail addresses: lany0001@ntu.edu.sg (Y. Lan), eycsoh@ntu.edu.sg (Y.C. Soh), egbhuang@ntu.edu.sg (G.-B. Huang).

0925-2312/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2009.02.013

there exist β_i, $\mathbf{a}_i$ and b_i such that

$$f_L(\mathbf{x}_j)=\sum_{i=1}^{L}\beta_i G(\mathbf{a}_i,b_i,\mathbf{x}_j)=\mathbf{t}_j,\quad j=1,\cdots,N, \tag{1}$$

where $\mathbf{a}_i$ and b_i are the learning parameters of the hidden nodes, β_i is the output weight, and $G(\mathbf{a}_i,b_i,\mathbf{x}_j)$ denotes the output of the ith hidden node with respect to the input $\mathbf{x}_j$. When using additive hidden node, $G(\mathbf{a}_i,b_i,\mathbf{x}_j)=g(\mathbf{a}_i\cdot\mathbf{x}_j+b_i), b_i\in R$, where $\mathbf{a}_i$ is the input weight vector, b_i is the bias of the ith hidden node, and $\mathbf{a}_i\cdot\mathbf{x}_j$ denotes the inner product of the two. When using RBF hidden node, $G(\mathbf{a}_i,b_i,\mathbf{x}_j)=g(b_i\|\mathbf{x}_j-\mathbf{a}_i\|), b_i\in R^+$, where $\mathbf{a}_i$ and b_i are the center and impact width of the ith RBF node, and R^+ indicates the set of all positive real values.

Assume the network has L hidden nodes and the data $\aleph=\{(\mathbf{x}_i,\mathbf{t}_i)|\mathbf{x}_i\in\mathbf{R}^n,\mathbf{t}_i\in\mathbf{R}^m,i=1,\ldots,N\}$ presents to the network sequentially (one-by-one or chunk-by-chunk). There are two phases in OS-ELM algorithm, an initialization phase and a sequential phase. In the initialization phase, $rank(\mathbf{H}_0)=L$ is required to ensure that OS-ELM can achieve the same learning performance as ELM, where $\mathbf{H}_0$ denotes the hidden output matrix for initialization phase. It means the number of training data required in the initialization phase N_0 has to be equal to or greater than L, i.e. $N_0\geqslant L$. And if $N_0=N$, OS-ELM is the same as batch ELM. Hence, ELM can be seen as a special case of OS-ELM when all the data present in one iteration.

Initialization phase: A small chunk of training data is used to initialize the learning, $\aleph_0=\{(\mathbf{x}_i,\mathbf{t}_i)\}_{i=1}^{N_0}$ from the given training set $\aleph=\{(\mathbf{x}_i,\mathbf{t}_i)|\mathbf{x}_i\in\mathbf{R}^n,\mathbf{t}_i\in\mathbf{R}^m,i=1,\ldots,N\}$, and $N_0\geqslant L$.

(a) Randomly assign the input parameters: for additive hidden nodes, parameters are input weights $\mathbf{a}_i$ and bias b_i; for RBF hidden nodes, parameters are center $\mathbf{a}_i$ and impact factor b_i; $i=1,\ldots,L$.

(b) Calculating the initial hidden layer output matrix $\mathbf{H}_0$

$$\mathbf{H}_0=\begin{bmatrix}G(\mathbf{a}_1,b_1,\mathbf{x}_1) & \cdots & G(\mathbf{a}_L,b_L,\mathbf{x}_1)\\ \vdots & \cdots & \vdots\\ G(\mathbf{a}_1,b_1,\mathbf{x}_{N_0}) & \cdots & G(\mathbf{a}_L,b_L,\mathbf{x}_{N_0})\end{bmatrix}_{N_0\times L} \tag{2}$$

(c) Estimating the initial output weight $\beta^{(0)}$
We have $\mathbf{T}_0=[\mathbf{t}_1,\ldots,\mathbf{t}_{N_0}]^T_{N_0\times m}$, and the problem is to minimize $\|\mathbf{H}_0\beta-\mathbf{T}_0\|$. From [1], we know that $\mathbf{H}^\dagger=(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T$. The solution to minimize $\|\mathbf{H}_0\beta-\mathbf{T}_0\|$ is given by $\beta^{(0)}=\mathbf{P}_0\mathbf{H}_0^T\mathbf{T}_0$, where $\mathbf{P}_0=(\mathbf{H}_0^T\mathbf{H}_0)^{-1}$, and $\mathbf{K}_0=\mathbf{H}_0^T\mathbf{H}_0=\mathbf{P}_0^{-1}$.

(d) Set $k=0$. (k: a parameter indicates the number of chunks of data that is presented to the network.)

Sequential learning phase: Present the $(k+1)$th chunk of new observations,

$$\aleph_{k+1}=\{(\mathbf{x}_i,\mathbf{t}_i)\}_{i=(\sum_{j=0}^{k}N_j)+1}^{\sum_{j=0}^{k+1}N_j},$$

and N_{k+1} denotes the number of observations in the $(k+1)$th chunk.

(a) Compute the partial hidden layer output matrix $\mathbf{H}_{k+1}$

$$\mathbf{H}_{k+1}=\begin{bmatrix}G\left(\mathbf{a}_1,b_1,\mathbf{x}_{\left(\sum_{j=0}^{k}N_j\right)+1}\right) & \cdots & G\left(\mathbf{a}_L,b_L,\mathbf{x}_{\left(\sum_{j=0}^{k}N_j\right)+1}\right)\\ \vdots & \cdots & \vdots\\ G\left(\mathbf{a}_1,b_1,\mathbf{x}_{\sum_{j=0}^{k+1}N_j}\right) & \cdots & G\left(\mathbf{a}_L,b_L,\mathbf{x}_{\sum_{j=0}^{k+1}N_j}\right)\end{bmatrix}_{N_{k+1}\times L} \tag{3}$$

(b) Calculate the output weight $\beta^{(k+1)}$
We have $\mathbf{T}_{k+1}=[\mathbf{t}_{(\sum_{j=0}^{k}N_j)+1},\ \cdots,\ \mathbf{t}_{\sum_{j=0}^{k+1}N_j}]^T_{N_{k+1}\times m}$. And

$$\mathbf{K}_{k+1}=\mathbf{K}_k+\mathbf{H}_{k+1}^T\mathbf{H}_{k+1} \tag{4}$$

$$\beta^{(k+1)}=\beta^{(k)}+\mathbf{K}_{k+1}^{-1}\mathbf{H}_{k+1}^T(\mathbf{T}_{k+1}-\mathbf{H}_{k+1}\beta^{(k)}) \tag{5}$$

From Eq. (5), we find that $\mathbf{K}_{k+1}^{-1}$ is used to compute $\beta^{(k+1)}$. In order to avoid calculating inverse in the recursive process, the Woodbury formula [14] is applied to modify the equations

$$\begin{aligned}\mathbf{K}_{k+1}^{-1}&=(\mathbf{K}_k+\mathbf{H}_{k+1}^T\mathbf{H}_{k+1})^{-1}\\&=\mathbf{K}_k^{-1}-\mathbf{K}_k^{-1}\mathbf{H}_{k+1}^T(\mathbf{I}+\mathbf{H}_{k+1}\mathbf{K}_k^{-1}\mathbf{H}_{k+1}^T)^{-1}\mathbf{H}_{k+1}\mathbf{K}_k^{-1}\end{aligned} \tag{6}$$

And $\mathbf{P}_{k+1}=\mathbf{K}_{k+1}^{-1}$, we modify the Eqs. (4) and (5) using (6):

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

$$\beta^{(k+1)}=\beta^{(k)}+\mathbf{P}_{k+1}\mathbf{H}_{k+1}^T(\mathbf{T}_{k+1}-\mathbf{H}_{k+1}\beta^{(k)}) \tag{8}$$

(c) Set $k=k+1$. Go to (a) in this sequential learning phase.

3. Ensemble of OS-ELM

EOS-ELM consists of many OS-ELM networks with same number of hidden nodes and same activation function for each hidden node. We have constructed P OS-ELM networks to form our EOS-ELM. All P OS-ELMs are trained with new data in each incremental step. The input parameters for each OS-ELM network are randomly generated and the output weights are obtained analytically based on the sequential arrived input data. Then we compute the average of the outputs of each OS-ELM network, which is the final output of the EOS-ELM. Assume the output of each OS-ELM network is $f^{(j)}(\mathbf{x}_i), j=1,\ldots,P$. Hence, we have

$$f(\mathbf{x}_i)=\frac{1}{P}\sum_{j=1}^{P}f^{(j)}(\mathbf{x}_i), \tag{9}$$

where $f(\mathbf{x}_i)$ is the output of the whole system with the input of $\mathbf{x}_i$.

We expect that EOS-ELM works better than individual OS-ELM network because the randomly generated parameters make each OS-ELM network in the ensemble distinct. Therefore, the OS-ELM networks composing the ensemble may have different adaptive capacity to the new data. When the data come into the ensemble network sequentially, some of OS-ELM networks may adapt faster and better to the new data than others. However, because for different incoming data, different OS-ELM networks can be the ones that have good adaptation, the EOS-ELM should be seen as a whole system. The system could avoid the cases when the individual OS-ELM network could not adapt well to the new data, which make the final result of individual OS-ELM network worst in the round of simulation as compared to other rounds. Statistically speaking, the population mean is always closer to the expectation than the elements in the population, which implies that the results obtained by EOS-ELM are always fluctuating in a smaller range as compared to the results obtained by single OS-ELM network. When P becomes larger, the mean value $f^{(j)}(\mathbf{x}_i)$ is closer to the expectation of $f^{(j)}(\mathbf{x}_i)$. Therefore, we may conclude that the EOS-ELM could be more stable when P is larger. However, when P increases, the computation time also increases and the network becomes more complex. Considering the computation time, complexity and stability of the network, we decide to conduct the selection of parameter P among 5, 10, 15, 20, 25, and 30. The simulation results are all presented in the following section.

When $N_0=N$, EOS-ELM becomes an ensemble of batch ELM networks [15]. Therefore, the ensemble of ELM proposed in [15] can be seen as a special case of EOS-ELM when all the training data are available at one time.