文章编号: 1001-0920(2017)10-1887-07 DOI: 10.13195/j.kzyjc.2016.0943

一种基于增量加权平均的在线序贯极限学习机算法

张明洋†, 闻英友, 杨晓陶, 赵 宏

(1. 东北大学 计算机科学与工程学院，沈阳 110819；
2. 东软公司 软件架构新技术国家重点实验室，沈阳 110179)

摘 要: 针对在线序贯极限学习机(OS-ELM)对增量数据学习效率低、准确性差的问题, 提出一种基于增量加权平均的在线序贯极限学习机(WOS-ELM)算法. 将算法的原始数据训练模型残差与增量数据训练模型残差进行加权作为代价函数,推导出用于均衡原始数据与增量数据的训练模型,利用原始数据来弱化增量数据的波动,使在线极限学习机具有较好的稳定性,从而提高算法的学习效率和准确性. 仿真实验结果表明, 所提出的WOS-ELM算法对增量数据具有较好的预测精度和泛化能力.

关键词: 单隐层前馈型神经网络；在线序贯极限学习机；加权平均；增量；代价函数

中图分类号: TP273 **文献标志码:** A

An incremental weighted average based online sequential extreme learning machine algorithm

ZHANG Ming-yang†, WEN Ying-you, YANG Xiao-tao, ZHAO Hong

(1. School of Computer Science and Engineering，Northeastern University，Shenyang 110819，China；2. State Key Laboratory of Software Architecture，Neusoft Corporation，Shenyang 110179，China)

Abstract: Considering the problem that online sequential extreme learning machine(OS-ELM) has low efficiency and accuracy when processing incremental data, an online sequential extreme learning machine algorithm based on incremental weighted average(WOS-ELM) is proposed. The principle of this algorithm is weighting model residuals trained with raw data and ones trained with incremental data, using the function got as the cost function of this algorithm, deducing a training model which is able to balance raw data and incremental data, and using raw data to weaken the fluctuation of incremental data. With this principle, the algorithm can raise stability, learning efficiency and accuracy of the OS-ELM. The result of a simulation experiment shows that the proposed algorithm has a higher predicting accuracy and better ability of generalization compared with other algorithms.

Keywords: single-hidden layer feedforward neural networks；online sequential extreme learning machine；weighted average；incremental；cost function

0 引 言

人工神经网络能够通过模仿生物神经网络的结构和功能,对输入和输出间复杂的非线性、统计性关系进行建模,探索数据间关系,具有较好的拟合能力和泛化能力,实现函数逼近、系统辨识等功能,在过去的几十年里得到了广泛的研究[1]. 近年来,极限学习机(ELM)[2]作为一种新的人工神经网络模型被提出. 该学习方法无需调整任何参数,通过隐含层输出矩阵的广义逆矩阵计算输出层权值,在保证网络具有良好泛化性能的同时,极大地提高了神经网络的学习速度,避免了传统的单隐层前馈型神经网络(SLFNs)普遍存在训练效率低、学习能力不强等问题[3]. 然而,对于成批次数据的训练,ELM不得不同时重复训练原有数据和新数据,训练效率低下. 为了满足训练数据成批次到达即在线学习的需求,Liang等[4]提出了在线序贯极限学习机,实现了逐个或逐块地对新增数据进行学习,相比于原始ELM在训练速度上有了较大提升. 由于在线序贯极限学习机训练速度快、准确率高,被应用于实时数据挖掘等方面,如:时间序列的预测[5-6]、股票市场的预测[7]、大数据挖掘[8]等.

近年来,在线序贯极限学习机的研究主要集中在训练增量数据时提高模型的稳定性等方面. 在线

收稿日期: 2016-07-23；**修回日期:** 2016-09-22.

基金项目: 国家863计划项目(2015AA016005)；国家自然科学基金项目(61402096, 61173153, 61300196).

作者简介: 张明洋(1989−), 男, 博士生, 从事机器学习、大数据挖掘的研究；赵宏(1954−), 男, 教授, 博士生导师, 从事计算机网络安全与信息安全、分布式多媒体、大数据挖掘等研究.

†通讯作者. E-mail: zmyzyc1988@126.com

序贯极限学习机(OS-ELM)算法虽然在训练速度上较原始ELM算法有很大提高,对于增量训练数据仍然无法取得较好的训练效果,预测精度相对较差. 对此,Lan等[9]提出了整体形式的在线序贯极限学习机(EOS-ELM)算法,通过将样本数据代入多个OS-ELM进行训练生成多个OS-ELM训练模型,将多个模型预测结果综合起来作为最终结果以提高预测的稳定性. Zhao等[10]考虑到数据的时间先后关系提出了带有遗忘机制的在线序贯极限学习机(FOS-ELM)算法,先将多个滑动窗口时间内的数据利用OS-ELM训练模型,再将多个模型预测结果综合起来作为最终预测结果,并通过实验验证了用于短期预测具有较好的效果.

本文在上述工作的基础上,首先针对ELM算法的代价函数进行改造,将原始数据训练模型残差与增量数据训练模型残差进行加权作为代价函数,推导出结合原始数据与增量数据的新的输出权重公式;然后,为满足OS-ELM的训练过程,将推导出的输出的权重公式重新表示为增量数据训练形式,并对权重公式按时间顺序进行扩展,提出加权的在线序贯极限学习机(WOS-ELM)算法;最后,对WOS-ELM算法按照集合形式进行扩展,提出整体形式的加权的在线序贯极限学习机(EWOS-ELM)算法. 将上述各算法用在典型的UCI数据集上进行对比实验,所得结果表明,WOS-ELM算法和EWOS-ELM算法具有更好的泛化能力和预测稳定性.

1 在线序贯极限学习机

1.1 极限学习机

极限学习机是近几年提出的一种新型单隐层前馈神经网络(SLFNs)的学习方法. 该方法在保证神经网络具有良好泛化性能的同时,极大地提高了前向神经网络的学习速度.

对于极限学习机,假设有N个不同的训练样本$(\boldsymbol{x}_j,\boldsymbol{t}_j)\in\boldsymbol{R}^n\times\boldsymbol{R}^m$. 其中:$\boldsymbol{x}_j$是$n\times1$维的输入向量,$\boldsymbol{t}_j$是$m\times1$维的目标向量. 若该神经网络包含$\tilde{N}$个隐层节点,则ELM可表示为

$$f_{\tilde{N}}(\boldsymbol{x}_j)=\sum_{i=1}^{\tilde{N}}\boldsymbol{\beta}_iG(\boldsymbol{\alpha}_i,b_i,\boldsymbol{x}_j)=\boldsymbol{t}_j,\quad j=1,2,\cdots,N. \tag{1}$$

其中:$G(\boldsymbol{\alpha}_i,b_i,\boldsymbol{x}_j)$是第$i$个隐层节点关于输入向量$\boldsymbol{x}_j$的激活函数,$\boldsymbol{\alpha}_i\in\boldsymbol{R}^n$是输入节点与第$i$个隐层节点的连接权值向量,$b_i\in\boldsymbol{R}$是第$i$个隐层节点的偏置,$\boldsymbol{\beta}_i\in\boldsymbol{R}^m$是第$i$个隐层节点与输出节点的连接权值向量. 式(1)可改写为矩阵形式

$$\boldsymbol{H}\boldsymbol{\beta}=\boldsymbol{T}. \tag{2}$$

其中

$$\boldsymbol{H}(\boldsymbol{\alpha}_1,\cdots,\boldsymbol{\alpha}_{\tilde{N}},b_1,\cdots,b_{\tilde{N}},\boldsymbol{x}_1,\cdots,\boldsymbol{x}_N)=\begin{bmatrix}G(\boldsymbol{\alpha}_1,b_1,\boldsymbol{x}_1)&\cdots&G(\boldsymbol{\alpha}_{\tilde{N}},b_{\tilde{N}},\boldsymbol{x}_1)\\ \vdots&\ddots&\vdots\\ G(\boldsymbol{\alpha}_1,b_1,\boldsymbol{x}_N)&\cdots&G(\boldsymbol{\alpha}_{\tilde{N}},b_{\tilde{N}},\boldsymbol{x}_N)\end{bmatrix}_{N\times\tilde{N}},$$

$$\boldsymbol{\beta}=\begin{bmatrix}\boldsymbol{\beta}_1^{\mathrm{T}}\\ \vdots\\ \boldsymbol{\beta}_{\tilde{N}}^{\mathrm{T}}\end{bmatrix}_{\tilde{N}\times m},\ \boldsymbol{T}=\begin{bmatrix}\boldsymbol{t}_1^{\mathrm{T}}\\ \vdots\\ \boldsymbol{t}_N^{\mathrm{T}}\end{bmatrix}_{N\times m},$$

$\boldsymbol{H}$称为网络的隐层节点输出矩阵.

ELM算法首先根据先验知识设定网络隐层节点类型和节点数$\tilde{N}$,节点类型可为可加性节点或径向基函数(RBF)类型节点,随机生成$\tilde{N}$个$\{(\boldsymbol{\alpha}_i,b_i)\}$对,结合输入向量$\{\boldsymbol{x}_j\}_{i=1}^N$生成$\boldsymbol{H}$,而后可求得输出权重矩阵$\hat{\boldsymbol{\beta}}$为

$$\hat{\boldsymbol{\beta}}=\boldsymbol{H}^{\dagger}\boldsymbol{T}. \tag{3}$$

其中$\boldsymbol{H}^{\dagger}$为$\boldsymbol{H}$的Moore-Penrose广义逆[12],对其可采用正交投影法、迭代法和奇异值分解法等方法[14]进行求解. 若矩阵$(\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H})$非奇异,根据正交投影法$\boldsymbol{H}^{\dagger}$可表示为$\boldsymbol{H}^{\dagger}=(\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H})^{-1}\boldsymbol{H}^{\mathrm{T}}$,则式(3)可表示为

$$\hat{\boldsymbol{\beta}}=(\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H})^{-1}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{T}, \tag{4}$$

然后利用式(2)对数据进行预测.

1.2 在线序贯极限学习机

为了满足训练数据成批次到达即在线学习的需求,Liang等[4]提出了在线极限学习机. 在线极限学习机递推计算公式如下:

$$\boldsymbol{P}_{k+1}=\boldsymbol{P}_k-\boldsymbol{P}_k\boldsymbol{H}_{k+1}^{\mathrm{T}}(\boldsymbol{I}+\boldsymbol{H}_{k+1}\boldsymbol{P}_k\boldsymbol{H}_{k+1}^{\mathrm{T}})^{-1}\boldsymbol{H}_{k+1}\boldsymbol{P}_k, \tag{5}$$

$$\boldsymbol{\beta}^{(k+1)}=\boldsymbol{\beta}^{(k)}+\boldsymbol{P}_{k+1}\boldsymbol{H}_{k+1}^{\mathrm{T}}(\boldsymbol{T}_{k+1}-\boldsymbol{H}_{k+1}\boldsymbol{\beta}^{(k)}). \tag{6}$$

其中

$$\boldsymbol{P}_{k+1}=\boldsymbol{K}_{k+1}^{-1}, \tag{7}$$

$$\boldsymbol{K}_{k+1}=\boldsymbol{K}_k+\boldsymbol{H}_{k+1}^{\mathrm{T}}\boldsymbol{H}_{k+1}, \tag{8}$$

$$\boldsymbol{T}_{k+1}=\begin{bmatrix}\boldsymbol{t}^{\mathrm{T}}_{(\sum_{j=0}^{k}N_j)+1}\\ \vdots\\ \boldsymbol{t}^{\mathrm{T}}_{\sum_{j=0}^{k+1}N_j}\end{bmatrix}_{N_{k+1}\times m}, \tag{9}$$

$$\boldsymbol{H}_{k+1}=\begin{bmatrix} G\Big(\boldsymbol{\alpha}_1,b_1,\boldsymbol{x}_{(\sum\limits_{j=0}^{k}N_j)+1}\Big) & \cdots & G\Big(\boldsymbol{\alpha}_{\tilde{N}},b_{\tilde{N}},\boldsymbol{x}_{(\sum\limits_{j=0}^{k}N_j)+1}\Big)\\ \vdots & \ddots & \vdots \\ G\Big(\boldsymbol{\alpha}_1,b_1,\boldsymbol{x}_{\sum\limits_{j=0}^{k+1}N_j}\Big) & \cdots & G\Big(\boldsymbol{\alpha}_{\tilde{N}},b_{\tilde{N}},\boldsymbol{x}_{\sum\limits_{j=0}^{k+1}N_j}\Big) \end{bmatrix}_{N_{k+1}\times\tilde{N}} \tag{10}$$

变量初始值为

$$\boldsymbol{K}_0=\boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{H}_0, \tag{11}$$

$$\boldsymbol{\beta}^{(0)}=\boldsymbol{K}_0^{-1}\boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{T}_0, \tag{12}$$

$$\boldsymbol{T}_0=\begin{bmatrix}\boldsymbol{t}_1^{\mathrm{T}}\\ \vdots\\ \boldsymbol{t}_{N_0}^{\mathrm{T}}\end{bmatrix}_{N_0\times m}, \tag{13}$$

$$\boldsymbol{H}_0=\begin{bmatrix} G(\boldsymbol{\alpha}_1,b_1,\boldsymbol{x}_1) & \cdots & G(\boldsymbol{\alpha}_{\tilde{N}},b_{\tilde{N}},\boldsymbol{x}_1)\\ \vdots & \ddots & \vdots\\ G(\boldsymbol{\alpha}_1,b_1,\boldsymbol{x}_{N_0}) & \cdots & G(\boldsymbol{\alpha}_{\tilde{N}},b_{\tilde{N}},\boldsymbol{x}_{N_0}) \end{bmatrix}_{N_0\times\tilde{N}}. \tag{14}$$

通过上述过程即可完成在线序贯极限学习机训练,然后利用式(2)进行预测.

1.3　在线序贯极限学习机衍生型

为了提高OS-ELM训练模型的稳定性和预测的准确性,各种基于OS-ELM的衍生算法被相继提出,其中较为典型的是Lan等[9]提出的EOS-ELM算法和Zhao等[10]提出的FOS-ELM算法. 下面针对这两种算法进行简单介绍.

EOS-ELM算法相当于在任意$K+1$时刻建立P个包含相同个数同样类型的隐层节点的OS-ELM,对每个OS-ELM分别应用式(5)和(6)进行训练,然后分别利用式(2)计算数据的预测值,并按照下式计算最终预测值:

$$\boldsymbol{T}=\frac{1}{P}\sum_{j=1}^{P}(\boldsymbol{H}\boldsymbol{\beta})_j. \tag{15}$$

FOS-ELM算法相当于在OS-ELM算法的基础上,将多个滑动窗口数据块利用OS-ELM进行训练,对每个OS-ELM分别应用下式进行训练:

$$\boldsymbol{P}_{k+1}^{(r)}=\boldsymbol{P}_k^{(r)}-\boldsymbol{P}_k^{(r)}\begin{bmatrix}-\boldsymbol{H}_{k-s+1}^{(r)}\\ \boldsymbol{H}_{k+1}^{(r)}\end{bmatrix}^{\mathrm{T}}\Big(\boldsymbol{I}+\begin{bmatrix}\boldsymbol{H}_{k-s+1}^{(r)}\\ \boldsymbol{H}_{k+1}^{(r)}\end{bmatrix}\times\boldsymbol{P}_k^{(r)}\begin{bmatrix}-\boldsymbol{H}_{k-s+1}^{(r)}\\ \boldsymbol{H}_{k+1}^{(r)}\end{bmatrix}^{\mathrm{T}}\Big)^{-1}\begin{bmatrix}\boldsymbol{H}_{k-s+1}^{(r)}\\ \boldsymbol{H}_{k+1}^{(r)}\end{bmatrix}\boldsymbol{P}_k^{(r)}, \tag{16}$$

$$\boldsymbol{\beta}^{(r)}(k+1)=\boldsymbol{\beta}^{(r)}(k)+\boldsymbol{P}_{k+1}^{(r)}\begin{bmatrix}-\boldsymbol{H}_{k-s+1}^{(r)}\\ \boldsymbol{H}_{k+1}^{(r)}\end{bmatrix}^{\mathrm{T}}\times\Big(\begin{bmatrix}\boldsymbol{T}_{k-s+1}^{(r)}\\ \boldsymbol{T}_{k+1}^{(r)}\end{bmatrix}-\begin{bmatrix}\boldsymbol{H}_{k-s+1}^{(r)}\\ \boldsymbol{H}_{k+1}^{(r)}\end{bmatrix}\boldsymbol{\beta}^{(r)}(k)\Big); \tag{17}$$

然后分别利用式(2)计算数据预测值,并按下式计算最终预测值:

$$f_{\tilde{N}}(\boldsymbol{Z}_{k+2})=\frac{1}{P}\sum_{r=1}^{P}f_{\tilde{N}}^{(r)}(\boldsymbol{Z}_{k+2}). \tag{18}$$

2　WOS-ELM算法

极限学习机在对增量数据进行训练时模型的稳定性较差,预测的准确性较低. 本文将在增量数据训练模型的基础上结合原来训练好的模型,对其进行加权平均作为预测模型. 设原训练样本为$(\boldsymbol{X}_{\mathrm{pre}},\boldsymbol{T}_{\mathrm{pre}})$,增量训练样本为$(\boldsymbol{X}_{\mathrm{inc}},\boldsymbol{T}_{\mathrm{inc}})$,新训练样本为$(\boldsymbol{X}_{\mathrm{new}},\boldsymbol{T}_{\mathrm{new}})$,则

$$[\boldsymbol{X}_{\mathrm{new}}\ \ \boldsymbol{T}_{\mathrm{new}}]=\begin{bmatrix}\boldsymbol{X}_{\mathrm{pre}} & \boldsymbol{T}_{\mathrm{pre}}\\ \boldsymbol{X}_{\mathrm{inc}} & \boldsymbol{T}_{\mathrm{inc}}\end{bmatrix}, \tag{19}$$

对应训练样本的激活函数矩阵可表示为

$$\boldsymbol{H}_{\mathrm{new}}=\begin{bmatrix}\boldsymbol{H}_{\mathrm{pre}}\\ \boldsymbol{H}_{\mathrm{inc}}\end{bmatrix}. \tag{20}$$

定义模型的代价函数为

$$\begin{aligned}\boldsymbol{E}=&\frac{1}{2}\lambda_{\mathrm{new}}\|\boldsymbol{H}_{\mathrm{new}}\boldsymbol{\beta}-\boldsymbol{T}_{\mathrm{new}}\|^2+\\ &\frac{1}{2}\lambda_{\mathrm{pre}}\|\boldsymbol{H}_{\mathrm{new}}\boldsymbol{\beta}-\hat{\boldsymbol{T}}_{\mathrm{pre}}\|^2=\\ &\frac{1}{2}\lambda_{\mathrm{new}}(\boldsymbol{H}_{\mathrm{new}}\boldsymbol{\beta}-\boldsymbol{T}_{\mathrm{new}})^{\mathrm{T}}(\boldsymbol{H}_{\mathrm{new}}\boldsymbol{\beta}-\boldsymbol{T}_{\mathrm{new}})+\\ &\frac{1}{2}\lambda_{\mathrm{pre}}(\boldsymbol{H}_{\mathrm{new}}\boldsymbol{\beta}-\hat{\boldsymbol{T}}_{\mathrm{pre}})^{\mathrm{T}}(\boldsymbol{H}_{\mathrm{new}}\boldsymbol{\beta}-\hat{\boldsymbol{T}}_{\mathrm{pre}}).\end{aligned} \tag{21}$$

其中: $\lambda_l(l=\mathrm{new},\mathrm{pre})$为权重,范围为$\lambda_l\in[0,1]$区间,且$\sum\limits_{l=\mathrm{new},\mathrm{pre}}\lambda_l=1$;$\hat{\boldsymbol{T}}_{\mathrm{pre}}$为添加增量数据前的训练样本的输出值与增量数据使用原来模型利用式(2)计算的预测值的集合,即

$$\hat{\boldsymbol{T}}_{\mathrm{pre}}=\begin{bmatrix}\boldsymbol{T}_{\mathrm{pre}}\\ \hat{\boldsymbol{T}}_{\mathrm{inc}}\end{bmatrix}=\begin{bmatrix}\boldsymbol{T}_{\mathrm{pre}}\\ \boldsymbol{H}_{\mathrm{inc}}\boldsymbol{\beta}_{\mathrm{pre}}\end{bmatrix}. \tag{22}$$

求式(21)代价函数最小值$\min\boldsymbol{E}$,即是对代价函数求关于$\boldsymbol{\beta}$的偏导数,有

$$d\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{\beta}} = \lambda_{\rm new}\boldsymbol{H}_{\rm new}^{\rm T}(\boldsymbol{H}_{\rm new}\boldsymbol{\beta}-\boldsymbol{T}_{\rm new})+ \lambda_{\rm pre}\boldsymbol{H}_{\rm new}^{\rm T}(\boldsymbol{H}_{\rm new}\boldsymbol{\beta}-\hat{\boldsymbol{T}}_{\rm pre}). \tag{23}$$

令式(23)为零,可求得$\boldsymbol{\beta}$值为

$$\boldsymbol{\beta} = (\boldsymbol{H}_{\rm new}^{\rm T}\boldsymbol{H}_{\rm new})^{-1}\times[\lambda_{\rm new}\boldsymbol{H}_{\rm new}^{\rm T}\boldsymbol{T}_{\rm new}+ \lambda_{\rm pre}\boldsymbol{H}_{\rm new}^{\rm T}\hat{\boldsymbol{T}}_{\rm pre}]. \tag{24}$$

式(24)即为模型的输出权重,结合式(2),即可对数据进行预测.

对于式(24)的λ_l,考虑两种极端情况:当$\lambda_{\rm pre}=0$时,$\lambda_{\rm new}=1,\boldsymbol{\beta}=(\boldsymbol{H}_{\rm new}^{\rm T}\boldsymbol{H}_{\rm new})^{-1}\boldsymbol{H}_{\rm new}^{\rm T}\boldsymbol{T}_{\rm new}$,即是由新样本数据按照原始ELM算法训练的模型;当$\lambda_{\rm pre}=1$时,$\lambda_{\rm new}=0,\boldsymbol{\beta}=(\boldsymbol{H}_{\rm new}^{\rm T}\boldsymbol{H}_{\rm new})^{-1}\boldsymbol{H}_{\rm new}^{\rm T}\hat{\boldsymbol{T}}_{\rm pre}$,结合式(6)对其进行处理可转换为$\boldsymbol{\beta}=\boldsymbol{\beta}_{\rm pre}+\boldsymbol{P}_{\rm new}\boldsymbol{H}_{\rm inc}^{\rm T}(\hat{\boldsymbol{T}}_{\rm inc}-\boldsymbol{H}_{\rm inc}\boldsymbol{\beta}_{\rm pre})$,其中$\hat{\boldsymbol{T}}=\boldsymbol{H}_{\rm inc}\boldsymbol{\beta}_{\rm pre}$,因此$\boldsymbol{\beta}=\boldsymbol{\beta}_{\rm pre}$,即是由原样本数据按照原始ELM算法训练的模型. 由此可以看出,λ的大小可以用来衡量新样本数据训练的模型和原样本数据训练的模型在最终模型中的各自权重. $\boldsymbol{\lambda}_{\rm new}$值越大,新样本数据在最终模型中的比重越大,对新样本数据的学习度越高,最终模型越趋向于新训练模型;$\boldsymbol{\lambda}_{\rm pre}$值越大,最终模型越趋向于原训练模型.

在上述模型的训练过程中,对于增量形式样本数据仍需对全部样本数据进行训练,学习效率较低. 为适应在线学习需求, 推导其增量递归形式,式(24)可转化为

$$\boldsymbol{\beta}= \left(\begin{bmatrix}\boldsymbol{H}_{\rm pre}\\ \boldsymbol{H}_{\rm inc}\end{bmatrix}^{\rm T}\begin{bmatrix}\boldsymbol{H}_{\rm pre}\\ \boldsymbol{H}_{\rm inc}\end{bmatrix}\right)^{-1}\left(\lambda_{\rm new}\begin{bmatrix}\boldsymbol{H}_{\rm pre}\\ \boldsymbol{H}_{\rm inc}\end{bmatrix}^{\rm T}\times \begin{bmatrix}\boldsymbol{T}_{\rm pre}\\ \boldsymbol{T}_{\rm inc}\end{bmatrix}+\lambda_{\rm pre}\begin{bmatrix}\boldsymbol{H}_{\rm pre}\\ \boldsymbol{H}_{\rm inc}\end{bmatrix}^{\rm T}\begin{bmatrix}\boldsymbol{T}_{\rm pre}\\ \hat{\boldsymbol{T}}_{\rm inc}\end{bmatrix}\right). \tag{25}$$

根据Woodbury公式[11]并结合式(6)可求得

$$\boldsymbol{\beta}= \lambda_{\rm new}(\boldsymbol{\beta}_{\rm pre}+\boldsymbol{P}_{\rm new}\boldsymbol{H}_{\rm inc}^{\rm T}(\boldsymbol{T}_{\rm inc}-\boldsymbol{H}_{\rm inc}\boldsymbol{\beta}_{\rm pre}))+ \lambda_{\rm pre}(\boldsymbol{\beta}_{\rm pre}+\boldsymbol{P}_{\rm new}\boldsymbol{H}_{\rm inc}^{\rm T}(\hat{\boldsymbol{T}}_{\rm inc}-\boldsymbol{H}_{\rm inc}\boldsymbol{\beta}_{\rm pre})). \tag{26}$$

由于$\hat{\boldsymbol{T}}=\boldsymbol{H}_{\rm inc}\boldsymbol{\beta}_{\rm pre}$,式(26)可转化为

$$\boldsymbol{\beta}=\boldsymbol{\beta}_{\rm pre}+\lambda_{\rm new}\boldsymbol{P}_{\rm new}\boldsymbol{H}_{\rm inc}^{\rm T}(\boldsymbol{T}_{\rm inc}-\boldsymbol{H}_{\rm inc}\boldsymbol{\beta}_{\rm pre}). \tag{27}$$

将式(27)按照递归形式表示为

$$\boldsymbol{\beta}^{(k+1)}= \boldsymbol{\beta}^{(k)}+\lambda_{\rm new}\boldsymbol{P}_{k+1}\boldsymbol{H}_{k+1}^{\rm T}(\boldsymbol{T}_{k+1}-\boldsymbol{H}_{k+1}\boldsymbol{\beta}^{(k)}). \tag{28}$$

至此,完成了对WOS-ELM基本算法的推导. 式(28)包括第k和$k+1$两个时刻的样本数据,考虑到可以将WOS-ELM的数据扩展到更早时刻,将$(L+1)$个时刻数据作为一个滑动窗口联合构造其代价函数,有

$$\boldsymbol{E}= \frac{1}{2}\lambda_0\|\boldsymbol{H}\boldsymbol{\beta}-\boldsymbol{T}_{\rm new}\|^2+\frac{1}{2}\sum_{l=1}^{L}\lambda_l\|\boldsymbol{H}\boldsymbol{\beta}-\hat{\boldsymbol{T}}_{{\rm pre}l}\|^2. \tag{29}$$

其中:$\lambda_l\geqslant 0,\ l=0,1,\cdots,L$且$\sum_{l=0}^{L}\lambda_l=1$;$\hat{\boldsymbol{T}}_{{\rm pre}l}$的定义同式(22);$L\geqslant 0,(L+1)$为选取增量样本数据的窗口长度. 类似计算,可求得

$$\boldsymbol{\beta}^{(k+1)}=\lambda_{k+1}(\boldsymbol{\beta}^{(k)}+\boldsymbol{P}_{k+1}\boldsymbol{H}_{k+1}^{\rm T}(\boldsymbol{T}_{k+1}-\boldsymbol{H}_{k+1}\boldsymbol{\beta}^{(k)}))+\sum_{l=k-L+1}^{k}(\lambda_l\boldsymbol{\beta}^{(l)}). \tag{30}$$

以上是$(L+1)$阶WOS-ELM算法对于历史增量样本数据按照滑动窗口方式训练模型的推导. $(L+1)$阶WOS-ELM可以构造类似于EOS-ELM的集合表达形式(EWOS-ELM),EWOS-ELM由P个具有相同隐层节点个数$\tilde{N}$和相同激活函数$G(\boldsymbol{\alpha},b,\boldsymbol{x})$的WOS-ELM构成,其中第$r$个WOS-ELM的输出权重可表示为

$$\boldsymbol{\beta}_{k+1}^{(r)}=\lambda_{k+1}(\boldsymbol{\beta}_{k}^{(r)}+\boldsymbol{P}_{k+1}\boldsymbol{H}_{k+1}^{\rm T}(\boldsymbol{T}_{k+1}-\boldsymbol{H}_{k+1}\boldsymbol{\beta}_{k}^{(r)}))+\sum_{l=k-L+1}^{k}(\lambda_l\boldsymbol{\beta}_{l}^{(r)}). \tag{31}$$

在预测阶段,结合式(15),可构造类似于EOS-ELM的用于计算最终预测值的等式

$$\boldsymbol{T}=\frac{1}{P}\sum_{r=1}^{P}(\boldsymbol{H}\boldsymbol{\beta}^{(r)}). \tag{32}$$

其中:$\boldsymbol{H}$为待预测数据代入第r个WOS-ELM模型所求得的$\boldsymbol{H}$矩阵,$\boldsymbol{\beta}^{(r)}$为第r个WOS-ELM模型的输出权重. 图1给出了EWOS-ELM算法流程.

下面结合图1的算法流程介绍EWOS-ELM算法的实现步骤.

Step 1: 设置各训练参数λ_l、$\tilde{N}$、L、P;

Step 2: 按照式(11) ~ (14)设置各变量初始值;

Step 3: 按照式(5)、(31)、(7) ~ (10)对增量数据进行训练,生成第r个WOS-ELM模型;

Step 4: 保留第r个模型的长度为L的滑动窗口的输出权重集$\{\boldsymbol{\beta}^{(l)}\}_{l=k-L+1}^{k}$;

Step 5: $r=r+1$,若$r<P$,则按照Step 3和Step 4对第r个模型进行训练;

Step 6: 按照式(32)计算模型的最终预测值.

以上为EWOS-ELM算法的实现步骤,其中,对于

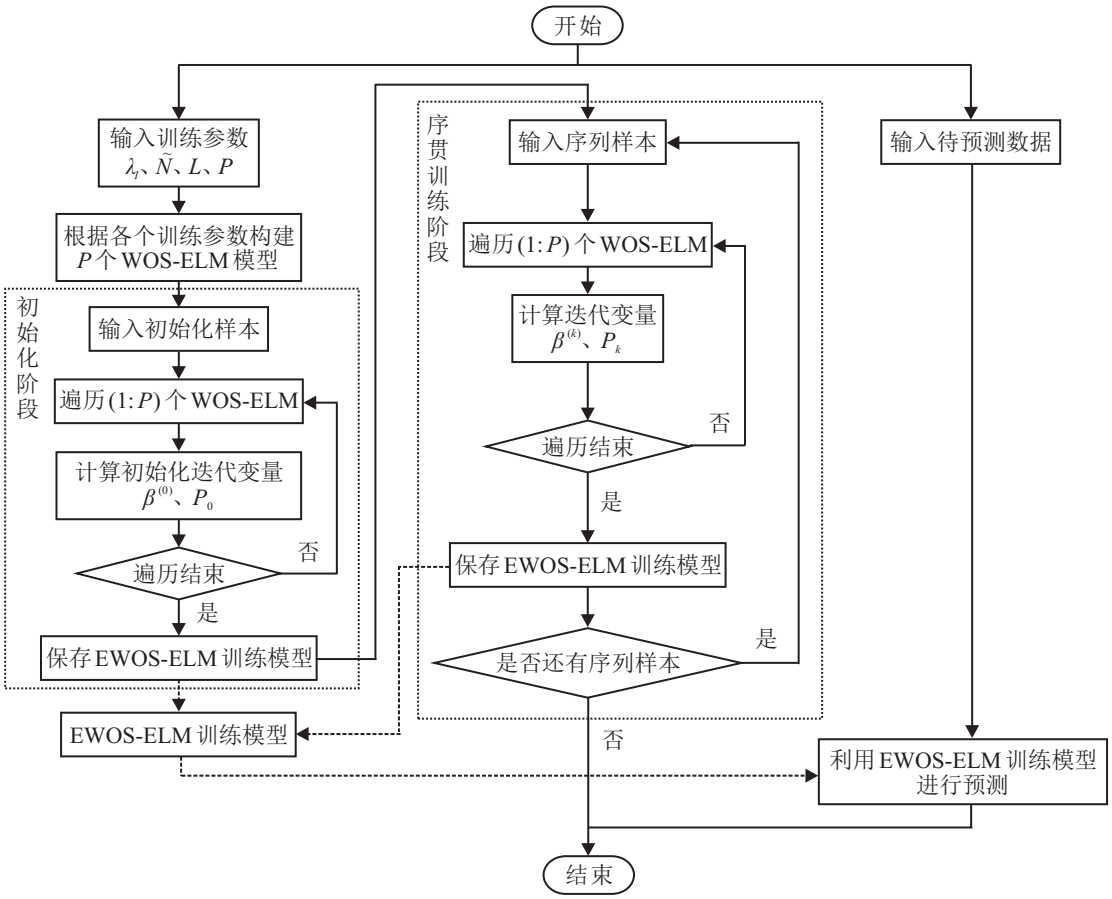

图 1　EWOS-ELM算法流程

各训练参数的设置可通过交叉验证进行确定,具体过程可参考仿真及结果分析部分相关内容. 本文的WOS-ELM与OS-ELM相比,除输出权重计算过程之外,其他计算过程完全相同,而输出权重计算过程本文算法与OS-ELM算法的时间开销基本相同,仅需额外计算L次矩阵加法操作;在空间开销方面,本文算法需额外开辟L个空间用于存储历史数据,通过加权平均的方式,利用历史数据的稳定性降低增量数据的误差造成的训练模型的波动,从而提高模型的学习效率和预测准确性.

3　仿真及结果分析

为验证本文所提出WOS-ELM算法的有效性,利用UCI真实数据集 (https://archive.ics.uci.edu/ml/) 对OS-ELM、EOS-ELM、FOS-ELM、WOS-ELM和EWOS-ELM进行分类和回归测试. 6组典型真实数据集分别是: Auto mpg、Boston housing、CCS、Image segment、Monks、Satellite image. 本文所有仿真均在同一系统环境下运行,系统运行环境为PC(CPU 3.30 GHz, 4.00 GB RAM),操作系统为Windows 7,仿真软件为Matlab R2012 a. 实验数据集分为训练数据集和测试数据集,具体描述如表1所示.

表 1　数据集描述

数据集	训练样本数	测试样本数	特征数	问题类型
Auto mpg	196	196	7	回归
Boston housing	253	253	13	回归
CCS	515	515	8	回归
Image segment	1 500	810	18	分类
Monks	124	432	6	分类
Satellite image	4 435	2 000	36	分类

在测试实验中,对样本数据中特征不完整记录进行剔除,并对输入数据和输出数据进行归一化处理:对于回归样本数据集,将输入数据归一化到$[-1, 1]$区间,相应的输出数据归一化至$[0, 1]$区间;对于分类样本数据集,将输入数据归一化到$[-1, 1]$区间,相应的输出数据按不同种类编号.

Sigmoid函数作为ELM网络的激活函数,对OS-ELM使用5-折交叉验证[13,9]确定隐层节点个数. 设置WOS-ELM的阶数为2阶,隐层节点个数与OS-ELM相同. 针对历史权重参数$\lambda_{\rm pre}$,按照每次增加0.05,遍历从0.1到1.0的所有值,训练WOS-ELM学习模型. 对于回归样本数据,计算训练数据和预测数据对应的均方根误差(RMSE);对于分类样本数据,计算训练数据和预测数据对应的准确率. 进行20次实验,计算20次实验结果的平均值和标准差(SD),选择

RMSE平均值最小值或准确率平均值最大值对应的$\lambda_{\rm pre}$权重值作为WOS-ELM对应样本集的参数. 图2所示为Boston housing样本数据在Sigmoid激活函数下不同的权重$\lambda_{\rm pre}$对应的误差曲线.

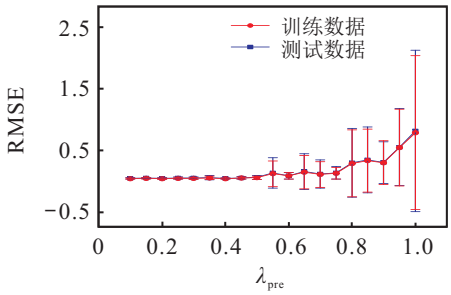

图 2 Boston housing数据集算法$\lambda_{\rm pre}$误差曲线

如图2所示, Boston housing样本数据对应WOS-ELM模型的$\lambda_{\rm pre}$权重值选取0.1时, RMSE获得最小值, 因此, 该样本数据对应的WOS-ELM算法的$\lambda_{\rm pre}$权重参数选取0.1. 同理, 可以测得其他样本数据对应WOS-ELM模型的$\lambda_{\rm pre}$权重参数值. $\lambda_{\rm pre}$权重参数值确定后, 设置EWOS-ELM算法中$\lambda_{\rm pre}$权重参数与WOS-ELM相同, 并针对EOS-ELM、FOS-ELM和EWOS-ELM三种算法所包含的神经网络个数P, 按照每次增加5, 遍历从5到40的所有值, 训练对应的3种学习模型, 计算训练数据、预测数据对应的RMSE或准确率, 进行20次实验, 计算20次实验结果的平均值和标准差, 选择RMSE平均值最小值或准确率最大值对应的神经网络个数P值作为EOS-ELM、FOS-ELM和EWOS-ELM对应样本集的参数.

通过上述方式设置OS-ELM、WOS-ELM、EOS-ELM、FOS-ELM和EWOS-ELM各算法的隐层节点数(# Nodes)、神经网络数(# Networks)、λ值等各参数, 对上述5种ELM算法针对表1中的6种数据集进行20次实验, 将20次实验结果的平均值作为最终结果. 表2和表3分别为回归、分类数据集应用以上5种算法的结果比较.

表2为通过UCI真实数据集验证5种在线极限学习机基于回归问题的算法结果的对比. 通过对3组回归数据集的实验结果对比可知, 本文所提出的WOS-ELM比原始OS-ELM算法对于回归问题的精度有了明显提高, 同时扩展的EWOS-ELM算法较EOS-ELM和FOS-ELM算法对于回归问题的精度也明显提高; 而WOS-ELM和EWOS-ELM算法的训练时间与OS-ELM和EOS-ELM的训练时间相似.

表3为通过UCI真实数据集验证5种在线极限学习机基于分类问题的算法结果的对比. 与表2的结果类似, 本文所提出的WOS-ELM的训练和测试精度均高于原始OS-ELM算法, 同时扩展的EWOS-ELM算法的精度也高于EOS-ELM和FOS-ELM算法; 而WOS-ELM和EWOS-ELM算法的训练时间与OS-ELM和EOS-ELM的训练时间相似.

通过对表2和表3基于回归和分类问题的算法结果的对比可知, 本文所提出的WOS-ELM和扩展的EWOS-ELM相比于OS-ELM、EOS-ELM和FOS-ELM算法, 对于分类和回归问题均有明显提高.

表 2 回归问题的训练和测试结果对比

数据集	算法	# Nodes	# Networks	λ	训练时间	RMSE或准确率		标准差	
						训练	测试	训练	测试
Auto mpg	OS-ELM	25	–	–	0.0156	0.0791	0.0941	0.0209	0.2697
	WOS-ELM	25	–	0.15	0.0546	0.0180	0.0207	0.0444	0.0431
	EOS-ELM	25	15	–	0.6754	0.0319	0.0358	0.0793	0.0837
	FOS-ELM	25	15	–	0.6692	0.0253	0.0316	0.0538	0.0583
	EWOS-ELM	25	15	0.15	0.6700	0.0111	0.0126	0.0228	0.0232
Boston housing	OS-ELM	30	–	–	0.0468	0.0200	0.0237	0.0042	0.0046
	WOS-ELM	30	–	0.10	0.0624	0.0197	0.0231	0.0038	0.0043
	EOS-ELM	30	5	–	0.3049	0.0155	0.0172	0.0198	0.0192
	FOS-ELM	30	5	–	0.2964	0.0136	0.0159	0.0121	0.0137
	EWOS-ELM	30	5	0.10	0.3096	0.0103	0.0120	0.0019	0.0023
CCS	OS-ELM	100	–	–	0.2184	0.0029	0.0045	0.0003	0.0003
	WOS-ELM	100	–	0.10	0.3822	0.0028	0.0044	0.0007	0.0011
	EOS-ELM	100	15	–	1.3891	0.0027	0.0041	0.0036	0.0057
	FOS-ELM	100	15	–	1.4032	0.0023	0.0035	0.0023	0.0040
	EWOS-ELM	100	15	0.10	1.3533	0.0016	0.0025	0.0003	0.0003

表 3　分类问题的训练和测试结果对比

数据集	算法	# Nodes	# Networks	λ	训练时间	RMSE或准确率		标准差	
						训练	测试	训练	测试
Image segmentation	OS-ELM	180	–	–	0.0538	0.9501	0.9403	0.0301	0.0269
	WOS-ELM	180	–	0.15	0.0530	0.9578	0.9452	0.0287	0.0363
	EOS-ELM	180	25	–	0.9063	0.9673	0.9508	0.0306	0.0444
	FOS-ELM	180	25	–	0.9984	0.9619	0.9514	0.0335	0.0375
	EWOS-ELM	180	25	0.15	1.0480	0.9690	0.9527	0.0243	0.0430
Satellite image	OS-ELM	400	–	–	3.3470	0.9019	0.8760	0.0362	0.0349
	WOS-ELM	400	–	0.15	3.8259	0.9107	0.8734	0.0353	0.0346
	EOS-ELM	400	25	–	86.3551	0.9122	0.8857	0.0122	0.0393
	FOS-ELM	400	25	–	97.2588	0.9195	0.8716	0.0214	0.0390
	EWOS-ELM	400	25	0.15	87.8808	0.9117	0.8816	0.0085	0.0389
monks-1	OS-ELM	80	–	–	0.0109	0.8294	0.7380	0.0108	0.0514
	WOS-ELM	80	–	0.10	0.0249	0.8681	0.7496	0.0179	0.0104
	EOS-ELM	80	20	–	0.1489	0.8770	0.8021	0.0274	0.0111
	FOS-ELM	80	20	–	0.1575	0.8681	0.8120	0.0300	0.0142
	EWOS-ELM	80	20	0.10	0.1731	0.9185	0.8580	0.0305	0.0109

4　结　论

本文针对在线序贯极限学习机对增量数据学习效率低、准确性差的问题,提出了一种基于增量加权平均的在线序贯极限学习机(WOS-ELM)算法. 构建了结合多个历史数据模型残差的代价函数,并根据代价函数推导出用于均衡原始数据与增量数据的训练模型. 增强了在线极限学习机的训练的稳定性,从而提高了算法的学习效率和准确性. 选取6组UCI典型数据集作为测试数据进行实验对比,结果表明,本文所提出的WOS-ELM算法稳定性较好,对增量数据具有较好的预测精度和泛化能力.

参考文献(References)

[1] 刘德荣, 李宏亮, 王鼎. 基于数据的自学习优化控制: 研究进展与展望[J]. 自动化学报, 2013, 39(11): 1858-1870.
(Liu D R, Li H L, Wang D. Data-based self-learning optimal control: Research progress and prospects[J]. Acta Automatica Sinica, 2013, 39(11): 1858-1870.)

[2] Huang G B, Zhu Q, Siew C K. Extreme learning machine: Theory and applications[J]. Neurocomputing, 2006, 70(1/2/3): 489-501.

[3] 王超, 王建辉, 顾树生,等. 改进式混合增量极限学习机算法[J]. 控制与决策, 2015, 30(11): 1981-1986.
(Wang C, Wang J H, Gu S S, et al. Improved hybrid incremental extreme learning machine algorithm[J]. Control and Decision, 2015, 30(11): 1981-1986.)

[4] Liang N Y, Huang G B, Saratchandran P, et al. A fast and accurate online sequential learning algorithm for feedforward networks[J]. IEEE Trans on Neural Networks, 2006, 17(6): 1411-1423.

[5] Rodolfo C C, Adriano L I O. An approach to handle concept drift in financial time series based on extreme learning machines and explicit drift detection[C]. The 2015 Int Joint Conf on Neural Networks. Killarney: IEEE, 2015: 1-8.

[6] 马超, 张英堂. 在线核极限学习机及其在时间序列预测中的应用[J]. 信息与控制, 2014, 43(5): 624-629.
(Ma C, Zhang Y T. Online kernel extreme learning machine and its application to time series prediction[J]. Information and Control, 2014, 43(5): 624-629.)

[7] Rodolfo C C, Adriano L I O. An autonomous trader agent for the stock market based on online sequential extreme learning machine ensemble[C]. The 2014 Int Joint Conf on Neural Networks. Beijing: IEEE, 2014: 1424-1431.

[8] Erik C, Huang G B, Liyanaarachchi L C K, et al. Extreme learning machines[J]. IEEE Intelligent Systems, 2013, 28(6): 30-59.

[9] Lan Y, Soh Y C, Huang G B. Ensemble of online sequential extreme learning machine[J]. Neurocomputing, 2009, 72(13/14/15): 3391-3395.

[10] Zhao J W, Wang Z H, Park D S. Online sequential extreme learning machine with forgetting mechanism[J]. Neurocomputing, 2012, 87(15): 79-89.

[11] Golub G H, Loan C F V. Matrix computations[M]. 4th ed. Baltimore: The Johns Hopkins University Press, 2013: 88-92.

[12] Rao C R, Mitra S K. Generalized inverse of a matrix and its applications[C]. Proc of the 6th Berkeley Symposium on Mathematical Statistics and Probability. Berkeley: University of California Press, 1972: 601-620.

[13] Kohavi R. A study of cross-validation and bootstrap for accuracy estimation and model selection[C]. Proc of the 14th Int Joint Conf on Artificial Intelligence. Montreal: Morgan Kaufmann Publishers Inc, 1995: 1137-1143.

[14] Golub G H, Van Loan C F. Matrix computations[M]. Baltimore: JHU Press, 2012: 69-75.

（责任编辑：李君玲）