



一种适用于物联网的在线 GP-ELM 算法

张 杰, 沈苏彬

(南京邮电大学 计算机学院, 南京 210046)

摘 要: 为满足物联网环境下边缘设备对机器学习算法准确、快速以及自适应产生参数的需求, 在 DE-ELM 的基础上提出一种在线的 GP-ELM 算法。通过改进结点增加方式, 在每次增加结点的同时添加结点统计和结点删除步骤, 提高训练速度, 同时保持算法的准确性。运用 Matlab 软件对图片分割、卫星图片分类、卫星 DNA 等数据集进行训练实验, 结果表明, 与 EI-ELM、D-ELM、EM-ELM 等算法相比, GP-ELM 算法在准确率、训练时间、模型大小和泛化能力等方面都表现出较好的学习性能。

关键词: 极限学习机; 动态学习; 物联网; 机器学习; 剪枝

开放科学(资源服务)标志码(OSID):



中文引用格式: 张杰, 沈苏彬. 一种适用于物联网的在线 GP-ELM 算法[J]. 计算机工程, 2020, 46(6): 314-320.

英文引用格式: ZHANG Jie, SHEN Subin. An online GP-ELM algorithm suitable for Internet of things[J]. Computer Engineering, 2020, 46(6): 314-320.

An Online GP-ELM Algorithm Suitable for Internet of Things

ZHANG Jie, SHEN Subin

(School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210046, China)

[Abstract] In order to meet the requirements of edge devices for accurate, fast and adaptive generated parameters of machine learning in the Internet of Things (IoT), this paper proposes an online GP-ELM algorithm based on the differential-evolution extreme learning machine, this paper proposes an online GP-ELM algorithm. The algorithm improves the way of adding nodes by carrying on node statistics and deleting nodes while everytime a node is added, so as to increase the training speed, while maintaining the accuracy of the algorithm. Matlab software is used to train the dataset image segmentation, satellite image classification, satellite DNA and conduct experiments. Results show that compared with EI-ELM, D-ELM, EM-ELM and other algorithms, GP-ELM algorithm has better performance in accuracy, training time, model size and generalization ability.

[Key words] Extreme Learning Machine (ELM); dynamic learning; Internet of Things (IoT); machine learning; pruning
DOI: 10.19678/j.issn.1000-3428.0055420

0 概述

自从极限学习机 (Extreme Learning Machine, ELM) 被提出以来, 由于其具有泛化能力强、训练时间短的优点, 受到了许多学者的关注。文献[1]证明了 ELM 的一致逼近性并将 ELM 直接应用于回归与多分类问题^[2]。为处理非平衡数据的学习问题, 文献[3]通过引入类别权值, 提出了加权极限学习机 (W-ELM)。文献[4]提出的在线序列极限学习机 (OS-ELM), 将 ELM 延伸至在线学习问题, 拓宽其实际应用范围。目前, ELM 在语音识别^[5]、电力系统^[6]等领域已得到初步应用。

作为一种新兴技术, ELM 在回归和分类应用中表现出良好的性能, 具有较高的运行速度和较好的通用性, 但为探索 ELM 在现实问题中的普遍适用性, 需要确定用于解决特定任务的最合适的网络架构。ELM 与批量学习方案^[7]以及在线顺序学习方案^[8]是以固定网络规模开发的, 搜索这种固定网络大小的常用方法是通过反复实验。该方法简单明了, 但计算成本高, 不能保证所选择的网络规模接近最优。文献[9]提出增量 ELM (I-ELM) 算法, 其中隐藏节点被逐一随机添加, 并且当添加新的隐藏节点时, 现有隐藏节点的输出权重被冻结, 证明了 I-ELM 的通用逼近能力。凸面 I-ELM (CI-ELM)^[10]采用另一

基金项目: 江苏省未来网络前瞻性研究项目 (BY20130951108)。

作者简介: 张 杰 (1994—), 男, 硕士, 主研方向为数据挖掘; 沈苏彬, 研究员、博士。

收稿日期: 2019-07-08 修回日期: 2019-09-18 E-mail: 977841468@qq.com

种源于巴伦凸优化概念的增量方法。当添加新的隐藏节点时, 这种方法允许适当地调整现有隐藏节点的输出权重。在这种情况下, CI-ELM 可以实现比 I-ELM 更快的收敛速度, 同时保留 I-ELM 的简单性。在 I-ELM 和 CI-ELM 的学习过程中, 一些新添加的隐藏节点仅在网络的最终输出中起很小的作用, 因此可能增加网络复杂性。为避免上述问题并获得更紧凑的网络架构, 文献[11]提出增强型 I-ELM (EI-ELM) 算法。在 EI-ELM 的每个学习步骤中, 随机生成若干隐藏节点, 并且仅将导致最大残余误差减少的节点添加到现有网络。此外, EI-ELM 还将 I-ELM 扩展到通用 SLFN。文献[12]在 SAP-ELM 算法的基础上, 结合 L2 正则化因子和奇异值分解 (SVD), 提出一种改进的 SAP-ELM (ImSAP-ELM) 算法以提高预测精度和数值稳定性。文献[13]对灵敏度进行修正, 提出更符合实际需求的神经网络剪枝算法 IS-ELM。

文献[14]提供一种误差最小化 ELM (EM-ELM) 方法。该方法允许逐个或逐组添加随机隐藏节点 (具有不同的大小)。此外, EM-ELM 在网络增长期间递进地更新输出权重, 降低了计算复杂度。然而, 在所有上述建设性 ELM 的实现中, 隐藏节点的数量随着学习进度和最终数量单调增加, 隐藏节点的数量相当于学习步骤。如果需要许多迭代步骤, 最终将获得大量隐藏节点, 而一些隐藏节点可能在网络输出中起很小的作用。文献[15]基于和声搜索算法优化极限学习机的方法, 结合了和声搜索算法和极限学习机两者的优势, 利用和声搜索算法调整极限学习机的输入权值和隐含层阈值, 避免了随机设定无效节点而导致预测效果不稳定、泛化能力较差等问题。文献[16]提出了一种新的隐含层节点的选择算法 SHN-ELM。通过对隐含层节点的选择, 提高了 ELM 算法的稳定性和分类准确率。文献[17]提出了以粒子群优化算法搜索 ELM 隐藏层最佳神经元个数, 用极限学习机模型的测试准确率作为粒子群优化算法适应值。文献[18]提出了一种 P-ELM 算法, 通过统计方法去除不相关或相关性较低的隐藏节点, 系统地确定学习过程中隐藏节点的数量, 但是这样的方法却无法自动地完成。文献[19]设计了一种基于 EFAST 的隐藏层节点剪枝算法 (FOS-ELM), 利用傅里叶敏感度测试方法对 OS-ELM 的隐藏节点进行分析, 能适用于剪枝后的网络参数调整算法。文献[20]针对隐藏节点数量影响 ELM 泛化性能的问题, 提出一种 SRM-ELM 算法。SRM-ELM 在结构风险最小化原则下, 建立隐藏节点数与泛化能力的关系函数, 利用 PSO 简单高效的全局搜索能力, 优化 ELM 的隐藏节点数, 避免了传统方法反复进行调节实验的繁琐。但是上述算法耗时比较长, 不适用于物联网的特性。

为此, 本文提出一种适合物联网的在线 GP-ELM 算法。该算法能够在添加多个节点的同时进行剪枝、删减节点, 以提高 ELM 的准确性。

1 极限学习机及其相关理论

1.1 极限学习机

在 ELM 中, 只需要预先确定隐藏神经元的数量, 而隐藏神经元的参数 (如 RBF 节点的中心和影响因子或附加节点的偏差和输入权重) 是随机分配的。因此, 给出如下 ELM 预测公式:

$$F_k(X) = \sum_{i=1}^k \beta_i H_i(X) = H(X) \beta \quad (1)$$

其中, $\beta = [\beta_1, \beta_2, \dots, \beta_k]$ 是连接隐藏层和输出层的输出权重的向量, $H(X) = [H_1(X), H_2(X), \dots, H_k(X)]$ 是隐藏层相对于样本 X 的输出。根据 Bartlett 理论, 对于训练误差较小的神经网络, 权重范数越小, 网络的泛化性能越好, 因此, 将训练误差与输出权重的范数最小化:

$$\text{Minimize: } \|H\beta - T\|^2 \text{ 和 } \|\beta\| \quad (2)$$

其中, H 是隐藏层的输出矩阵:

$$H = \begin{bmatrix} h(X_1) \\ \vdots \\ h(X_N) \end{bmatrix} = \begin{bmatrix} h_1(X_1) & \cdots & h_k(X_1) \\ \vdots & & \vdots \\ h_1(X_N) & \cdots & h_k(X_N) \end{bmatrix} \quad (3)$$

并且:

$$T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix} \quad (4)$$

将最小范数最小二乘法用于经典 ELM 的原始实现中, 即:

$$\beta = H^\dagger T \quad (5)$$

其中, H^\dagger 是 H 的 Moore-Penrose 广义逆, 即伪逆。

如果使用标准优化方法, 则基于约束优化的 ELM 可以写成:

$$\text{Minimize: } \frac{1}{2} \|\beta\|^2 + C \frac{1}{2}$$

受限于:

$$\xi = T - H\beta \quad (6)$$

根据 KKT 条件, 可以通过以下双重优化来解决:

$$L = \frac{1}{2} \|\beta\|^2 + C \frac{1}{2} \|\xi\|^2 - \alpha(\xi - T + H\beta) \quad (7)$$

其中, $\alpha_i \in \alpha = [\alpha_1, \alpha_2, \dots, \alpha_N]$ 对应于训练样本的拉格朗日乘数。KKT 最佳条件如下:

$$\frac{\partial L}{\partial \beta} = 0 \rightarrow \beta = H^T \alpha \quad (8)$$

$$\frac{\partial L}{\partial \xi} = 0 \rightarrow \alpha = \xi C \quad (9)$$

$$\frac{\partial L}{\partial \alpha} = 0 \rightarrow H\beta - T^T + \xi^T = 0 \quad (10)$$

根据训练集的大小可以实现不同的解决方案:

1) 训练样本数量很大的情况

如果训练数据 N 的数量非常大,如比特征空间 L 的维数大得多,即 $N \gg L$,则以下解决方案是优选的。

$$\beta = \left(\frac{1}{C} + H^T H \right)^{\dagger} H^T T \quad (11)$$

在这种情况下,ELM 的输出函数为:

$$F_L(X) = H(X) \left(\frac{1}{C} + H^T H \right)^{\dagger} H^T T \quad (12)$$

2) 训练样本数量少的情况

在这种情况下,通过将式(8)、式(9)代入式(10),可以等效地将式(11)、式(12)写为:

$$\left(\frac{1}{C} + HH^T \right) \alpha = T \quad (13)$$

由式(4)~式(8),有:

$$\beta = H^T \left(\frac{1}{C} + HH^T \right)^{-1} T \quad (14)$$

ELM 的输出方程为:

$$F_L(X) = H(X) \beta = H(X) H^T \left(\frac{1}{C} + HH^T \right)^{-1} T \quad (15)$$

1.2 EM-ELM 算法

EM-ELM 是一种简单而有效的算法,可以逐个或逐组地随机添加隐藏节点。在网络增长期间,基于误差最小化方法递增地更新输出权重。

给定一组训练数据 $\{(x_i, t_i)\}_{i=1}^N \subset R^d \times R$,如果网络的输出等于目标,则有:

$$f_n(x_j) = \sum_{i=1}^n \beta_i G(a_i, b_i, x_j) = t_j, j=1, 2, \dots, N \quad (16)$$

可以等效地以矩阵形式表示:

$$H\beta = T \quad (17)$$

其中:

$$H = \begin{pmatrix} G(a_1, b_1, x_1) & \cdots & G(a_n, b_n, x_1) \\ G(a_1, b_1, x_2) & \cdots & G(a_n, b_n, x_2) \\ \vdots & & \vdots \\ G(a_1, b_1, x_N) & \cdots & G(a_n, b_n, x_N) \end{pmatrix}_{N \times n}$$

$$\beta = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix}_{n \times 1}, T = \begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_n \end{pmatrix}_{N \times 1} \quad (18)$$

其中, H 被称为网络的隐藏层输出矩阵。在 ELM 中,随机生成隐藏层输出矩阵 H 。因此,训练 SLFN 仅相当于获得线性系统相对于输出权重向量 $\hat{\beta}$ 的解,即:

$$\hat{\beta} = H^{\dagger} T \quad (19)$$

其中, H^{\dagger} 是 H 的 Moore-Penrose 广义逆,即伪逆。

在 EM-ELM 中,给定一个目标误差 $\epsilon > 0$ 和一个先初始化好 n_0 个隐藏节点的前向神经网络 (SLFN)

$f_{n_0}(x_j) = \sum_{i=1}^{n_0} \beta_i G(a_i, b_i, x)$ 。由 H_1 表示该网络的隐藏层输出矩阵,然后根据式(19),可以通过计算输出权重矩阵 $\beta_1 = H_1^{\dagger} T$ 。

如果网络输出误差 $\|H_1 \beta_1 - T\| > \epsilon$,则新的隐藏节点 $\sigma n_0 = n_1 - n_0$ 被添加到现有 SLFN,并且新的隐藏层输出矩阵变为 $H_2 = [H_1, \sigma H_1]$ 。在这种情况下,与通过直接计算 Moore-Penrose 逆 H_2^{\dagger} 获得 $\beta_2 = H_2^{\dagger} T$ 的 ELM 不同,EM-ELM 提出了一种快速输出权重更新方法来计算 H_2^{\dagger} ,可知:

$$H_2^{\dagger} = (H_2^T H_2)^{-1} H_2^T = \left(\begin{bmatrix} H_1^T \\ \sigma H_1^T \end{bmatrix} \begin{bmatrix} H_1, \sigma H_1 \end{bmatrix} \right)^{-1} \begin{bmatrix} H_1^T \\ \sigma H_1^T \end{bmatrix} \quad (20)$$

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \left(\begin{bmatrix} H_1^T \\ \sigma H_1^T \end{bmatrix} \begin{bmatrix} H_1, \sigma H_1 \end{bmatrix} \right)^{-1} \quad (21)$$

根据 Schur 补充, H_2^{\dagger} 可以通过以下方式推导:

$$H_2^{\dagger} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} H_1^T \\ \sigma H_1^T \end{bmatrix} = \begin{bmatrix} A_{11} H_1^T + A_{12} \sigma H_1^T \\ A_{21} H_1^T + A_{22} \sigma H_1^T \end{bmatrix} = \begin{bmatrix} U_1 \\ D_1 \end{bmatrix} \quad (22)$$

其中:

$$D_1 = ((I - H_1 H_1^{\dagger}) \sigma H_1)^{\dagger}$$

$$U_1 = H_1^{\dagger} - H_1^{\dagger} \sigma H_1 D_1 \quad (23)$$

然后计算 β_2 :

$$\beta_2 = H_2^{\dagger} T = \begin{bmatrix} U_1 \\ D_1 \end{bmatrix} T \quad (24)$$

同样,输出权重可以逐步更新为:

$$\beta_{k+1} = H_{k+1}^{\dagger} T = \begin{bmatrix} U_k \\ D_k \end{bmatrix} T$$

$$D_k = ((I - H_k H_k^{\dagger}) \sigma H_k)^{\dagger}$$

$$U_k = H_k^{\dagger} - H_k^{\dagger} \sigma H_k D_k \quad (25)$$

1.3 在线序列学习机

训练数据可以在实际应用中逐块或逐个(块的特殊情况)到达。在线序列学习机(OS-ELM)算法针对在线学习案例,并在短时间内不断更新输出权重。

当新的采样数据到来时,ELM 的数学模型应该被修改为:

$$\begin{bmatrix} H \\ \delta H \end{bmatrix} \beta' = \begin{bmatrix} T \\ \delta T \end{bmatrix} \quad (26)$$

其中, δH 和 δT 是新生成的隐藏层输出矩阵,使用当前学习参数和由新获得的观测值组成的输出矩阵, β' 是修改后的输出权重矩阵。OS-ELM 算法分为两个阶段,即初始化阶段和顺序阶段。初始化阶段与普通 ELM 算法相同,而输出权重矩阵 β 将通过迭代方式在顺序阶段中更新。

OS-ELM 算法步骤如下:

步骤1(初始化阶段) 给定用于初始化学习的小块训练数据, $S_0 = \{(x_i, t_i) \mid x_i \in \mathbb{R}^n, t_i \in \mathbb{R}^m, i = 1, 2, \dots, N_0\}$ 来自整个训练集 S 。值得注意的是,初始化阶段中所需的训练数据的数量 n_0 应等于或大于隐藏节点的数量 \tilde{N} 。

1) 随机分配学习参数 a_j 和 $b_j (j = 1, 2, \dots, \tilde{N})$ 。

2) 计算初始隐藏层输出矩阵 H_0 :

$$H_0 = \begin{bmatrix} G(a_1, b_1, x_1) & \cdots & G(a_{\tilde{N}}, b_{\tilde{N}}, x_1) \\ \vdots & & \vdots \\ G(a_1, b_1, x_{N_0}) & \cdots & G(a_{\tilde{N}}, b_{\tilde{N}}, x_{N_0}) \end{bmatrix}$$

3) 计算初始输出权重 $\hat{\beta}^{(0)} = (H_0^T H_0)^{-1} H_0^T T_0$ 。

4) 设置 $k = 0$, 其中 k 是表示呈现给网络的数据块数量的索引。

步骤2(序列阶段) 给出第 $(k + 1)$ 个新观察结果:

$$S_{k+1} = \left\{ (x_i, t_i) \mid x_i \in \mathbb{R}^n, t_i \in \mathbb{R}^m, \right. \\ \left. i = \sum_{j=0}^k n_j + 1, \dots, \sum_{j=0}^{k+1} n_j \right\}$$

其中, n_k 表示 k 块中新获得的观察的数量。显然,可以得到 $N_0 = n_0$ 。

1) 计算部分隐藏层输出矩阵 h_{k+1} :

$$h_{k+1} = \begin{bmatrix} G(a_1, b_1, x_{N_{k+1}}) & \cdots & G(a_{\tilde{N}}, b_{\tilde{N}}, x_{N_{k+1}}) \\ \vdots & & \vdots \\ G(a_1, b_1, x_{N_{k+1}}) & \cdots & G(a_{\tilde{N}}, b_{\tilde{N}}, x_{N_{k+1}}) \end{bmatrix}$$

2) 计算输出权重矩阵 β^{k+1} :

$$\begin{cases} \hat{\beta}^{(k+1)} = \hat{\beta}^{(k)} + P_{k+1} h_{k+1}^T (t_{k+1} - h_{k+1} \hat{\beta}^{(k)}) \\ P_{k+1}^{-1} = P_k^{-1} + h_{k+1}^T h_{k+1} \end{cases}$$

其中, P_k 是更新矩阵, 并且 $P_0 = H_0^T H_0$ 。

设置 $k = k + 1$, 然后返回序列学习阶段。

2 GP-ELM 算法

2.1 算法描述

最早的 ELM 应用是预先设定好隐藏层节点的个数, 存在准确率不稳定等不足。该类算法主要有 2 种: 1) 逐渐增加一个或者多个节点, 这种算法将节点添加之后, 隐藏层的参数即固定, 这样一些不必要的节点或者贡献很小的节点会增加计算量; 2) 先设定一个较大的值作为隐藏层节点的个数, 根据第一次训练节点的权值删减一些节点, 然后再训练, 这种算法必定要设置一个非常大的值, 这对不同的数据的适应性就非常低, 因为一些数据只需要较少的

节点, 而有些则需要较多的节点。

本文算法是在 EM-ELM^[12] 算法模型的基础上, 固定每轮增加节点的个数, 增加了每轮对整体节点进行剪枝的过程, 能够保证节点的质量。本文算法结合上文两类算法的优点, 既能动态地根据数据来生成隐藏节点, 又能剔除那些没有起到作用的节点, 先生成固定数值节点, 在这些节点加入之后进行一次训练, 根据权值矩阵 β 来计算每个节点的贡献值:

1) 如果 β 是一维的, 则直接将这一列换算成比例。

2) 如果 β 是 n 维的, 则先将每一列换算成比例, 然后将每一行取平均值。

根据计算出的贡献值来选择删去哪些节点, 然后返回到增加节点的步骤, 如此循环, 直到达到一定的标准停止循环。

由于物联网下的数据是不断产生的, 因此在线训练数据更新参数又是非常有必要的, 所以本文算法在确定隐藏层结构后, 会根据新来的数据更新参数。

2.2 算法步骤

本文算法步骤如下:

步骤1 初始化阶段:

1) 随机生成隐藏节点的参数 $(a_1, b_1) \in \mathbb{R}^d \times \mathbb{R}$ 。

2) 计算隐藏层的输出矩阵:

$$H = \begin{bmatrix} G(a_1, b_1, x_1) \\ \vdots \\ G(a_1, b_1, x_N) \end{bmatrix}$$

3) 计算最优的输出权重:

$$\beta = H^+ T$$

其中, H^+ 为 H 的广义逆, T 为目标矩阵。于是, 得到初始函数 Ψ_1 以及相关的误差 E_1 :

$$\Psi_1(x) = \beta_1 G_1(x), E_1 = \|\Psi_1 - t\|$$

步骤2 隐藏节点上升并且删减(最大隐藏节点数 L_{\max} , 需求的错误率 ε):

1) 生成 M 个隐藏节点参数并将其加入到当前模型, 然后根据迭代式计算:

$$\beta_{k+1} = H_{k+1}^+ T = \begin{bmatrix} U_k \\ D_k \end{bmatrix} T$$

$$D_k = ((I - H_k H_k^+) \sigma H_k)^+$$

$$U_k = H_k^+ - H_k^+ \sigma H_k D_k$$

2) 计算出输出权重, 生成模型 $\Psi_n(x) = \beta_n G_n(x)$, 计算其误差 E_n , 如果 $L_n < L_{\max}$ 并且 $E_n > \varepsilon$, 置 $n = n + 1$, 进入步骤 3; 否则 $\Psi_n(x)$ 为当前标准模型。

3) 根据计算出的 β 计算每个节点的贡献值:

(1) 如果 β 是一维的, 则直接将这一列换算成比例。

(2) 如果 β 是 n 维的, 则先将每一列换算成比例, 然后将每一行取平均值。

4) 删除那些几乎不起作用的节点 (两种方案):

(1) 节点贡献值低于某一个值会被删除。

(2) 对贡献值进行排序, 贡献值最小的 m 个节点会被删除。

5) 转到步骤 1。

步骤 3 在线学习阶段:

设置 $r = 0$, 其中 k 是表示呈现给网络的数据块的数量的索引, 给出第 $(r + 1)$ 个新观察结果:

$$S_{r+1} = \left\{ (x_i, t_i) \mid x_i \in \mathbb{R}^n, t_i \in \mathbb{R}^m \right\}$$

$$i = \sum_{j=0}^r n_j + 1, \dots, \sum_{j=0}^{r+1} n_j \}$$

其中, n_r 表示 r 块中新获得的观察的数量。显然, 可以得到 $N_0 = n_0$ 。

1) 计算部分隐藏层输出矩阵 h_{r+1} :

$$h_{r+1} = \begin{bmatrix} G(a_1, b_1, x_{N_{r+1}}) & \cdots & G(a_{\tilde{N}}, b_{\tilde{N}}, x_{N_{r+1}}) \\ \vdots & & \vdots \\ G(a_1, b_1, x_{N_{r+1}}) & \cdots & G(a_{\tilde{N}}, b_{\tilde{N}}, x_{N_{r+1}}) \end{bmatrix}$$

2) 计算输出权重矩阵 β^{k+1} :

$$\begin{cases} \hat{\beta}^{(r+1)} = \hat{\beta}^{(r)} + P_{r+1} h_{r+1}^T (t_{r+1} - h_{k+1} \hat{\beta}^{(r)}) \\ P_{r+1}^{-1} = P_r^{-1} + h_{r+1}^T h_{r+1} \end{cases}$$

其中, P_r 是更新矩阵, 并且 $P_0 = H_0^T H_0$ 。

3) 如果还有数据进入, 则设置 $r = r + 1$, 然后返回序列学习阶段, 否则程序结束。

2.3 收敛性证明

定理 1 给定一个 SLFN, 令 H_1 作为有 L_0 个隐藏节点的 SLFN 隐藏层输出矩阵, 如果 $L_1 - L_0$ 个节点被加入当前模型, 新的隐藏层输出矩阵为 H_2 , 于是有:

$$E(H_2) = \min \|H_2 \beta_2 - T\| \leq E(H_1) = \|H_1 \beta_1 - T\|$$

证明: 根据前文的推理, 增加 $\sigma L_0 = L_1 - L_0$ 可以得到 $H_2 = [H_1 \sigma H_1]$, 其中:

$$\sigma H_1 = \begin{bmatrix} G(a_{L_0+1}, b_{L_0+1}, x_1) & \cdots & G(a_{L_1}, b_{L_1}, x_1) \\ \vdots & & \vdots \\ G(a_{L_0+1}, b_{L_0+1}, x_N) & \cdots & G(a_{L_1}, b_{L_1}, x_N) \end{bmatrix}$$

因为 β_2 是 $\min \|H_2 \beta_2 - T\|$ 最小二乘法的解, 所以有:

$$\begin{aligned} E(H_2) &= \min \|H_2 \beta_2 - T\| = \\ &= \min \|[H_1 \sigma H_1] \beta_2 - T\| \leq \\ &= \min \|[H_1 \sigma H_1] [\beta_1^T 0]^T - T\| = \\ &= \min \|H_1 \beta_1 - T\| = E(H_1) \end{aligned}$$

定理 2 给定一个组具体的训练样本和任意的正数 ε , 存在一个正整数 k 使得 $E(H_k) = \|H_k \beta_k - T\| \leq \varepsilon$ 。

证明: 假设 $k = 0$, 并且 SLFN 有 L_0 个隐藏节点, 如果它对应的输出误差 $E(H_1)$ 小于等于 ε , 那么增长阶段结束; 否则, 不断地一个一个或者一组一组地增加隐藏节点, 在第 k 步增加了 σL_k 个节点, 直到 $H_{k+1} = [H_k \sigma H_k]$, 根据定理 1, $E(H_k)$ 随着 k 不断减小, 即 $E(H_1) \geq E(H_2) \geq \cdots \geq E(H_k)$, 然后根据方程解, 如果总节点数大于样本数, H_k 则变成可逆的, 因此 $E(H_k) = 0$, 所以, 就会存在一个正整数 k 使得 $E(H_k) = \|H_k \beta_k - T\| \leq \varepsilon$ 。

由上面的证明可以看出, 本文算法可以使得误差值达到指定小的值, 即算法是收敛的, 能够迭代终止。

3 实验与结果分析

3.1 实验环境

本文实验的实验平台为 Intel i7-6700 3.4 GHz, 16 GB 内存和 1 TB 硬盘的 PC, 实验在 Windows 10 系统上用 Matlab 2016(b) 实现。本文实验采用了实际应用的公开的数据集 Image Segment (图片分割)、Satellite Image (卫星图片分类) 和 DNA, 如表 1 所示。

表 1 实验所用数据集

Table 1 Datasets used in experiment

名称	特征数	类别数	训练集	测试集
Image Segment	19	7	1 500	810
satellite Image	36	6	3 217	3 218
DNA	180	3	2 000	1 186

3.2 实验结果对比

本文实验将提出的算法 GP-ELM 与 ELM 以及各种衍生版本 OS-ELM、EI-ELM、D-ELM、EM-ELM 进行了对比 (本文所有实验结果都是重复实验 10 次后的均值)。

3.2.1 各个数据 ELM 网络结构的确定

对每个数据集逐渐增加隐藏节点的个数, 如图 1 所示。

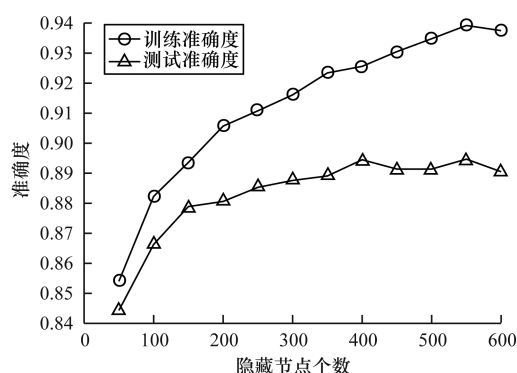


图 1 准确度随节点数增长过程

Fig. 1 Increasing process of accuracy with number of nodes

从图 1 可以看出,随着隐藏节点个数的增长,训练准确度和测试准确度都在逐步上升,测试准确度在 500 隐藏节点时达到了平稳状态,并且在 550 左右开始下降,因此,对于数据集 Satellite Image, ELM 的最优隐藏节点个数在 550 左右。对于其他两组数据集进行同样的实验得到了相应的节点个数。

3.2.2 与 ELM、OS-ELM 算法的对比

将本文算法与 ELM、OS-ELM 算法进行实验对比,结果如表 2 所示。从表 2 可以看出,本文算法在增加较少训练时间的代价下,首先完成了隐藏节点个数自动生成的任务,并且在准确度上有了一定的提升,同时需要更少的隐藏节点来完成。这是因为本文提出的算法在每次增加节点之后,对一些不重要的对结果影响特别小的那些隐藏节点都进行删除,这样,留下来的节点都是提取了非常重要的特征,对最后的结果有着很大的影响,所以本文提出的算法能用比较少的节点得到比较高的准确度。

表 2 GP-ELM 与 ELM、OS-ELM 算法性能对比

Table 2 Comparison of GP-ELM and ELM, OS-ELM algorithm performance

算法	数据集	训练时间/s	训练准确度/%	测试准确度/%	节点个数
ELM	Image Segment	0.267 9	98.43	95.56	450
	Satellite Image	0.531 3	93.61	89.38	550
	DNA	0.801 5	95.93	93.51	300
OS-ELM	Image Segment	0.473 5	98.27	95.29	450
	Satellite Image	0.434 4	93.52	89.26	550
	DNA	0.793 1	95.79	93.43	300
GP-ELM	Image Segment	1.187 5	99.07	95.37	385
	Satellite Image	1.796 8	93.67	89.31	494
	DNA	1.368 1	96.01	93.39	254

3.2.3 与 EI-ELM、D-ELM、EM-ELM 算法的对比

本文算法与 EI-ELM、D-ELM、EM-ELM 算法的性能对比如表 3 所示。从表 3 可以看出,本文算法具有更高的准确度,这是因为删去不重要的节点,使得本文算法生成的节点数量基本是最少的。另外,DP-ELM 在训练时间上相比其他算法要少,这是由于本文的算法并不是一个一个地增加节点,而是一块一块地增加,这样大幅节省了训练时间,同时,本文算法通过对原有的节点进行删除修剪,这样更有效地对 ELM 的结构进行组装,提高所选节点的质量,在得到较高准确度的同时节省了随机搜索而进行的大量迭代时间,保证了更少的节点。

表 3 GP-ELM 与 EI-ELM、D-ELM、EM-ELM 算法性能对比

Table 3 Comparison of GP-ELM and EI-ELM, D-ELM, EM-ELM algorithm performance

算法	数据集	训练时间/s	训练准确度/%	测试准确度/%	节点个数
GP-ELM	Image Segment	1.187 5	99.07	95.37	385
	Satellite Image	1.796 8	93.67	89.31	494
	DNA	1.368 1	96.01	93.39	254
EI-ELM	Image Segment	1.731 7	98.56	94.98	394
	Satellite Image	2.489 2	93.56	89.41	486
	DNA	2.584 1	95.33	93.48	261
D-ELM	Image Segment	4.773 5	98.79	95.03	386
	Satellite Image	5.831 9	93.61	89.25	521
	DNA	5.846 3	95.84	93.57	273
EM-ELM	Image Segment	1.406 2	98.73	95.18	431
	Satellite Image	2.750 0	93.64	88.92	561
	DNA	2.374 9	95.66	93.15	302

通过观察图 2 所示的迭代次数与被加入的隐藏节点个数的对比,可以发现 GP-ELM 是每次增加多个节点然后进行修剪,所以会很快地并以较少的节点完成迭代,可以看出本文算法具有较大的优势。

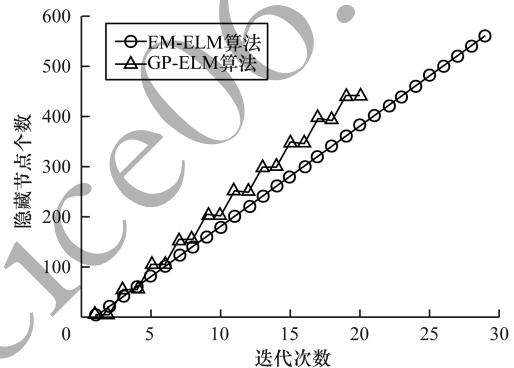


图 2 节点个数随迭代次数变化过程

Fig. 2 Changing process of number of nodes with number of iterations

4 结束语

为适应机器学习算法准确、快速以及自适应产生参数的需求,本文通过研究 ELM 网络结构的确定方法,结合 EM-ELM 算法节点增加的方式和广义逆的迭代公式,引入节点的修剪删除的阶段,实现达到根据数据本身自动地确定 ELM 网络结构的能力,并且在确定的过程中不断地删除贡献较低的节点,以保持网络的大小,提高 ELM 的准确性。实验结果表明,与 EM-ELM、D-ELM 和 EI-ELM 算法相比,DP-ELM 算法能够保证泛化能力,在准确率、训练时间、模型大小等方面都有着很好的表现,满足物联网下边缘设备对机器学习产生的应用需求。下一步将该算法拓展到真实的物联网场景下进行应用,以测试实际的效果与适用范围。

参考文献

- [1] HUANG G B, CHEN L, SIEW C K. Universal approximation using incremental constructive feedforward networks with random hidden nodes [J]. IEEE Transactions on Neural Networks, 2006, 17(4): 879-892.
- [2] HUANG Guangbin, ZHOU Hongming, DING Xiaojian, et al. Extreme learning machine for regression and multiclass classification [J]. IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics, 2012, 42(2): 513-529.
- [3] ZONG Weiwei, HUANG Guangbin, CHEN Yiqiang. Weighted extreme learning machine for imbalance learning [J]. Neurocomputing, 2013, 101: 229-242.
- [4] LIANG N Y, HUANG G B, SARATCHANDRAN P, et al. A fast and accurate online sequential learning algorithm for feedforward networks [J]. IEEE Transactions on Neural Networks, 2006, 17(6): 1411-1423.
- [5] LAN Y, HU Z J, SOH Y J, et al. An extreme learning machine approach for speaker recognition [J]. Neural Computing and Applications, 2013, 22(3/4): 417-425.
- [6] XU Yan, DAI Yuanyu, DONG Zhaoyang, et al. Extreme learning machine-based predictor for real-time frequency stability assessment of electric power systems [J]. Neural Computing and Applications, 2013, 22(3/4): 501-508.
- [7] HUANG G B, SIEW C K. Extreme learning machine; RBF network case [C]//Proceedings of the 8th IEEE Control, Automation, Robotics and Vision Conference. Washington D. C., USA: IEEE Press, 2004: 1029-1036.
- [8] LAN Y, SOH Y C, HUANG G B. Ensemble of online sequential extreme learning machine [J]. Neurocomputing, 2009, 72(13): 3391-3395.
- [9] HUANG Guangbin, LIANG Nanying, RONG Haijun, et al. On-line sequential extreme learning machine [C]//Proceedings of IASTED '05. Calgary, Canada: [s. n.], 2005: 125-132.
- [10] HUANG Guangbin, CHEN Lei. Convex incremental extreme learning machine [J]. Neurocomputing, 2007, 70(16): 3056-3062.
- [11] HUANG Guangbin, CHEN Lei. Enhanced random search based incremental extreme learning machine [J]. Neurocomputing, 2008, 71(16): 3460-3468.
- [12] DU Zhanlong, LI Xiaomin, XI Leiping, et al. Improved sensitivity-analysis based pruning extreme learning machine [J]. Control and Decision, 2016, 31(2): 249-255. (in Chinese)
杜占龙, 李小民, 席雷平, 等. 改进的灵敏度剪枝极限学习机 [J]. 控制与决策, 2016, 31(2): 249-255.
- [13] ZAI Huawei, CUI Licheng, ZHANG Weishi. Novel online adaptive algorithm of extreme learning machine based on improved sensitivity analysis [J]. Journal of Chinese Computer Systems, 2019, 40(7): 1386-1390. (in Chinese)
翟华伟, 崔立成, 张维石. 一种改进灵敏度分析的在线自适应极限学习机算法 [J]. 小型微型计算机系统, 2019, 40(7): 1386-1390.
- [14] FENG Guorui, HUANG Guangbin, LIN Qingbing, et al. Error minimized extreme learning machine with growth of hidden nodes and incremental learning [J]. IEEE Transactions on Neural Networks, 2009, 20(8): 1352-1357.
- [15] HUANG Qingbao, JIANG Chenglong, LIN Xiaofeng, et al. Optimization of extreme learning machine network based on harmony search algorithm [J]. Journal of Guangxi University (Natural Science Edition), 2018, 43(2): 517-524. (in Chinese)
黄清宝, 蒋成龙, 林小峰, 等. 基于和声搜索算法的极限学习机网络优化 [J]. 广西大学学报(自然科学版), 2018, 43(2): 517-524.
- [16] DENG Wanyu, ZHANG Shasha, LIU Guangda, et al. Extreme learning machine with selected hidden neurons [J]. Information Technology, 2018, 42(8): 1-3, 7. (in Chinese)
邓万宇, 张莎莎, 刘光达, 等. 极限学习机中隐含层节点选择研究 [J]. 信息技术, 2018, 42(8): 1-3, 7.
- [17] PAN Huaiqi, BI Yingzhou, PAN Yonghua. Optimal extreme learning machine based on particle swarm optimization algorithm [J]. Journal of Guangxi Teachers Education University (Natural Science Edition), 2018, 35(4): 49-53. (in Chinese)
潘怀奇, 闭应洲, 潘永华. 基于粒子群优化算法的最优极限学习机 [J]. 广西师范学院学报(自然科学版), 2018, 35(4): 49-53.
- [18] RONG H J, ONG Y S, TAN A Z, et al. A fast pruned-extreme learning machine for classification problem [J]. Neurocomputing, 2008, 72(1): 359-366.
- [19] DING Wangbin, WEI Shaohan, ZHANG Bixian. A online extreme learning machine node pruning method based on EFAST [J]. Journal of Sanming University, 2018, 35(4): 55-59. (in Chinese)
丁王斌, 魏少涵, 张碧仙. 基于 EFAST 的在线极限学习机节点剪枝方法 [J]. 三明学院学报, 2018, 35(4): 55-59.
- [20] XU Zhezhuang, HUANG Yanwei, LAI Dahu. Optimization for hidden nodes number of ELM based on approximate structure risk [J]. Computer Engineering, 2014, 40(9): 215-219, 224. (in Chinese)
徐哲壮, 黄宴委, 赖大虎. 基于近似结构风险的 ELM 隐层节点数优化 [J]. 计算机工程, 2014, 40(9): 215-219, 224.

编辑 索书志