神经网络极速学习方法研究

邓万宇[1)] 郑庆华[1)] 陈 琳[2)] 许学斌[1)]

[1)](西安交通大学电信学院计算机系智能网络与网络安全教育部重点实验室 西安 710049)

[2)](西安邮电学院计算机科学与技术系 西安 710061)

摘 要 单隐藏层前馈神经网络(Single-hidden Layer Feedforward Neural Network,SLFN)已经在模式识别、自动控制及数据挖掘等领域取得了广泛的应用,但传统学习方法的速度远远不能满足实际的需要,成为制约其发展的主要瓶颈.产生这种情况的两个主要原因是:(1)传统的误差反向传播方法(Back Propagation,BP)主要基于梯度下降的思想,需要多次迭代;(2)网络的所有参数都需要在训练过程中迭代确定.因此算法的计算量和搜索空间很大.针对以上问题,借鉴 ELM 的一次学习思想并基于结构风险最小化理论提出一种快速学习方法(RELM),避免了多次迭代和局部最小值,具有良好的泛化性、鲁棒性与可控性.实验表明 RELM 综合性能优于 ELM、BP 和 SVM.

关键词 极速学习机;正则极速学习机;支持向量机;结构风险;神经网络;最小二乘

中图法分类号 TP18 **DOI 号**: 10.3724/SP.J.1016.2010.00279

Research on Extreme Learning of Neural Networks

DENG Wan-Yu[1)] ZHENG Qing-Hua[1)] CHEN Lin[2)] XU Xue-Bin[1)]

[1)](*Ministry of Education Key Laboratory for Intelligent Networks and Network Security, Department of Computer Science and Technology, Xi'an Jiaotong University, Xi'an* 710049)

[2)](*Department of Computer Science and Technology, Xi'an University of Posts & Telecommunications, Xi'an* 710121)

Abstract SLFNs(Single-hidden Layer Feed forward Neural networks) have been widely applied in many fields including pattern recognition, automatic control, data mining etc. However, the traditional learning methods can not meet the actual needs due to two main reasons. Firstly, the traditional method is mainly based on gradient descent and it needs multiple iterations. Secondly, all of the network parameters need to be determined by iteration. Therefore, the computational complexity and searching space will increase dramatically. To solve the above problem, motivated by ELM's one-time learning idea, a novel algorithm called Regularized Extreme Learning Machine (RELM) based on structural risk minimization and weighted least square is proposed in this paper. The algorithm not only avoids a number of iterations and the local minimum, but also has better generalization, robustness and controllability than the original ELM. Additionally, experimental results have shown that RELM' overall performance is also better than BP and SVM.

Keywords extreme learning machine; regularized extreme learning machine; support vector machine; structural risk; neural network; least square

---

收稿日期:2009-07-15;最终修改稿收到日期:2009-09-27.本课题得到国家自然科学基金(60825202,60803079,60633020)、国家"八六三"高技术研究发展计划项目基金(2008AA01Z131)、国家科技支撑计划项目(2006BAK11B02,2006BAJ07B06,2008BAH26B02,2009BAH51B00)、中国科学院复杂系统与智能研究科学重点实验室开放基金资助项目(20080101)和陕西省教育厅科学研究计划项目(09JK717)资助.**邓万宇**,男,1979 年生,博士研究生,讲师,主要研究方向为机器学习、协作过滤、个性化服务.E-mail:dengwanyu@126.com.**郑庆华**,男,1969 年生,博士,教授,博士生导师,主要研究领域为智能化学习理论、网络安全.E-mail:qhzheng@xjtu.edu.cn.**陈 琳**,女,1977 年生,硕士,讲师,主要研究方向为机器学习、协作过滤、个性化服务.**许学斌**,男,1974 年生,硕士,工程师,主要研究方向为机器学习、模式识别.

1 引 言

单隐藏层前馈神经网络(Single-hidden Layer Feedforward Neural Network,SLFN)之所以能够在很多领域得到广泛应用,是因为它有很多优点:(1)具有很强的学习能力,能够逼近复杂非线性函数;(2)能够解决传统参数方法无法解决的问题.但另一方面缺乏快速学习方法,也使其很多时候无法满足实际需要.

对于SLFN的学习能力,很多文献分别从紧集(compact input sets)和有限集(infinite input sets)两种输入情况进行了深入讨论.Hornik研究表明:如果激励函数连续、有界且不是常量函数,那么SLFN能够在紧集情况下逼近任何连续函数[1];Leshno在Hornik基础上的进一步研究表明:使用非多项式激励函数的SLFN能够逼近任何连续函数[2].在实际应用中,神经网络的输入往往是有限集,对于有限集情况下SLFN的学习能力,Huang和Babri等进行了研究,结果表明:对于含有 N 个不同实例的有限集,一个具有非线性激励函数的SLFN最多只需 N 个隐藏层结点,就可以无误差地逼近这 N 个实例[3-4].这就是说,一个具有 N 个隐藏层结点的SLFN,即使输入权值随机取值,它也能够准确拟合 N 个不同的实例,更明确地讲就是:SLFN的学习能力只和隐藏层结点的数目有关,而和输入层的权值无关.虽然这一点对于提出一种新的学习算法很有启发,但并未引起研究者的注意,迭代调整的思想一直坚持到现在,很多算法都只是围绕这一思想进行技巧性的改进.不同于传统的学习方法,Huang基于以上研究结论为SLFN提出了一种称为极速学习机(Extreme Learning Machine,ELM)的学习方法[5]:设置合适的隐藏层结点数,为输入权和隐藏层偏差进行随机赋值,然后输出层权值通过最小二乘法得到.整个过程一次完成,无需迭代,与BP相比速度显著提高(通常10倍以上).

但是ELM是基于经验风险最小化原理,这可能会导致过度拟合问题[6].此外因为ELM不考虑误差的权重,当数据集中存在离群点时,它的性能将会受到严重影响[7].为了克服这些缺点,我们结合结构风险最小化理论以及加权最小二乘法对ELM算法进行改进,使得ELM在保持"快速"这一优势的前提下,泛化性能得到进一步的提高.

2 SLFN的统一模型

对于 N 个不同样本 $(\boldsymbol{x}_i,\boldsymbol{t}_i)$,其中 $\boldsymbol{x}_i=[x_{i1},x_{i2},\cdots,x_{in}]^{\mathrm{T}}\in R^n$, $\boldsymbol{t}_i=[t_{i1},t_{i2},\cdots,t_{im}]^{\mathrm{T}}\in R^m$,一个隐藏层结点数目为 $\tilde{N}$、激励函数为 $g(x)$ 的SLFN的统一模型为

$$\sum_{i=1}^{\tilde{N}}\boldsymbol{\beta}_i g_i(\boldsymbol{x}_j)=\sum_{i=1}^{\tilde{N}}\boldsymbol{\beta}_i g(\boldsymbol{a}_i\cdot\boldsymbol{x}_j+b_i)=\boldsymbol{t}_j,\ j=1,2,\cdots,N \tag{1}$$

其中 $\boldsymbol{a}_i=[a_{i1},a_{i2},\cdots,a_{in}]^{\mathrm{T}}$ 是连接第 i 个隐藏层结点的输入权值;b_i 是 i 个隐藏层结点的偏差(bias);$\boldsymbol{\beta}_i=[\beta_{i1},\beta_{i2},\cdots,\beta_{im}]^{\mathrm{T}}$ 是连接 i 个隐藏层结点的输出权值;$\boldsymbol{a}_i\cdot\boldsymbol{x}_j$ 表示 $\boldsymbol{a}_i$ 和 $\boldsymbol{x}_j$ 的内积.激励函数 $g(x)$ 可以是"Sigmoid"、"Sine"或"RBF"等.

上述 N 个方程的矩阵形式可写为

$$\boldsymbol{H}\boldsymbol{\beta}=\boldsymbol{T},$$

其中

$$\boldsymbol{H}(\boldsymbol{a}_1,\cdots,\boldsymbol{a}_{\tilde{N}},b_1,\cdots,b_{\tilde{N}},\boldsymbol{x}_1,\cdots,\boldsymbol{x}_N)=\begin{bmatrix}g(\boldsymbol{a}_1\cdot\boldsymbol{x}_1+b_1) & \cdots & g(\boldsymbol{a}_{\tilde{N}}\cdot\boldsymbol{x}_1+b_{\tilde{N}})\\ \vdots & \cdots & \vdots\\ g(\boldsymbol{a}_1\cdot\boldsymbol{x}_N+b_1) & \cdots & g(\boldsymbol{a}_{\tilde{N}}\cdot\boldsymbol{x}_N+b_{\tilde{N}})\end{bmatrix}_{N\times\tilde{N}},$$

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

$E(\boldsymbol{W})$ 表示期望值和实际值之间的误差平方和,问题求解就是寻找最优的权值 $\boldsymbol{W}=(\boldsymbol{a},\boldsymbol{b},\boldsymbol{\beta})$ 使代价函数 $E(\boldsymbol{W})$ 最小,其数学模型可表示为

$$\begin{aligned}&\arg\min_{\boldsymbol{W}=(\boldsymbol{a},\boldsymbol{b},\boldsymbol{\beta})}E(\boldsymbol{W})=\arg\min_{\boldsymbol{W}=(\boldsymbol{a},\boldsymbol{b},\boldsymbol{\beta})}\|\boldsymbol{\varepsilon}\|^2,\\ &\text{s.t.}\sum_{i=1}^{\tilde{N}}\boldsymbol{\beta}_i g(\boldsymbol{a}_i\cdot\boldsymbol{x}_j+b_i)-\boldsymbol{t}_j=\boldsymbol{\varepsilon}_j,\ j=1,2,\cdots,N\end{aligned} \tag{2}$$

其中 $\boldsymbol{\varepsilon}_j=[\varepsilon_{j1},\varepsilon_{j2},\cdots,\varepsilon_{jm}]$ 是第 j 个样本的误差.

为了方便讨论,在后文中将以一维输出($m=1$)为例进行研究,但所得结论仍适用于多维情况.

3 BP

由Rumelhart和McClelland提出的BP神经网络模型是目前应用最广泛的模型之一[8],BP训练方法是通过反向误差传播原理不断调整网络权值使得实际输出与期望输出之间的误差平方和达到最小或小于某个阈值.当 $\boldsymbol{H}$ 未知时,通常采用梯度下降法

迭代调整 $\boldsymbol{W}$：

$$\boldsymbol{W}_k=\boldsymbol{W}_{k-1}-\eta\frac{\partial E(\boldsymbol{W})}{\partial \boldsymbol{W}},$$

其中 η 代表学习速率.

基于梯度下降法的 BP 存在以下缺点：

(1) 训练速度慢. 因为需要多次的迭代，所以时间消耗很长.

(2) 参数选择很敏感，必须选取合适的 η 与 $\boldsymbol{W}$ 初值，才能取得理想的结果. 若 η 太小，算法收敛很慢，而 η 太大，算法不太稳定甚至不再收敛；

(3) 局部最小值. 由于 $E(\boldsymbol{W})$ 非凸，因此在下降过程中可能会陷入局部最小点，无法达到全局最小[9]；

(4) 过渡拟合. 在有限样本上训练时，仅以训练误差最小为目标的训练可能导致过渡拟合.

4　ELM

为了解决以上问题，Huang 基于以下定理为 SLFN 提出了 ELM 学习算法.

定理 1[5]. 对于任意 N 个不同样本 $(\boldsymbol{x}_i,\boldsymbol{t}_i)$，其中 $\boldsymbol{x}_i=[x_{i1},x_{i2},\cdots,x_{in}]^{\mathrm{T}}\in R^n$，$\boldsymbol{t}_i=[t_{i1},t_{i2},\cdots,t_{im}]^{\mathrm{T}}\in R^m$，$N$ 个隐藏层结点和一个任意区间无限可导的激活函数 $g:R\to R$，则 SLFN 在 $\boldsymbol{a}_i\in R^n$ 和 $b_i\in R$ 任意赋值的情况下，所形成的隐藏层矩阵 $\boldsymbol{H}$ 可逆，即方程组有精确解，代价函数 $E(\boldsymbol{W})=0$.

定理 2[5]. 给定任意 N 个不同样本 $(\boldsymbol{x}_i,\boldsymbol{t}_i)$，任意小误差 $e>0$，及在任意区间无限可导的激活函数 $g:R\to R$，总存在一个包含 $\widetilde{N}(\widetilde{N}\leq N)$ 个隐藏层结点的 SLFN，使得在 $\boldsymbol{a}_i\in R^n$ 和 $b_i\in R$ 任意取值情况下，误差 $E(\boldsymbol{W})\leq e$.

定理 1 和定理 2 的详细证明可参考文献[4-5,10]. 定理表明：只要隐含层结点数足够多，SLFN 就能在输入权随机赋值情况下逼近任何连续函数. 但为了使 SLFN 具有良好的泛化性能，通常 $\widetilde{N}\ll N$. 当输入权以随机赋值的方式确定后，所得隐藏层矩阵 $\boldsymbol{H}$ 便是一个确定的矩阵，因此训练 SLFN 就转化为计算 $\boldsymbol{H\beta}=\boldsymbol{T}$ 的最小二乘解问题. 关于 ELM 的细节请参考文献[5].

与 BP 相比 ELM 需要调整的参数只有隐含层结点个数 $\widetilde{N}$，目前虽没有精确估计 $\widetilde{N}$ 的方法，但 $\widetilde{N}\ll N$ 大大缩小了搜索范围，在实际应用中 $\widetilde{N}$ 可以通过交叉验证的方式确定. 在标准 UCI 数据集上的大量实验表明 ELM 训练速度快，泛化性能良好，但 ELM 仍有一些缺点：

(1) ELM 仅考虑经验风险，没有考虑到结构化风险，因此可能导致过度拟合问题；

(2) ELM 直接计算最小二乘解，用户无法根据数据集的特征进行微调，可控性差；

(3) 当数据集中存在离群点时，模型性能将会受到很大影响，鲁棒性较差.

为了克服这些缺点，我们把结构风险最小化理论以及加权最小二乘法引入到 ELM 中，提出一种正则极速学习机(Regularized Extreme Learning Machine，RELM).

5　正则极速学习机(RELM)

根据统计学理论可知，实际风险包括经验风险和结构风险两种成分[11]. 一个具有较好泛化性能的模型应该能权衡这两种风险，并取得最佳的折中. RELM 将同时考虑这两种风险因素，并通过参数 γ 调节两种风险的比例，RELM 的数学模型可表示为

$$\arg\min_{\boldsymbol{\beta}} E(\boldsymbol{W})=\arg\min_{\boldsymbol{\beta}}\left(\frac{1}{2}\|\boldsymbol{\beta}\|^2+\frac{1}{2}\gamma\|\boldsymbol{\varepsilon}\|^2\right),$$

$$\text{s.t.}\ \sum_{i=1}^{\widetilde{N}}\boldsymbol{\beta}_i g(\boldsymbol{a}_i\cdot\boldsymbol{x}_j+b_i)-\boldsymbol{t}_j=\boldsymbol{\varepsilon}_j,\ \ j=1,2,\cdots,N,$$

其中，误差的平方和 $\|\boldsymbol{\varepsilon}\|^2$ 代表经验风险；$\|\boldsymbol{\beta}\|^2$ 代表结构风险，它源于统计理论中边缘距离最大化原理[12-13]；而 γ 则是两种风险的比例参数，通过交叉验证的方式确定 γ 来获得两种风险的最佳折中点.

为了获得一个抗干扰模型，我们为不同样本的误差进行加权，$\|\boldsymbol{\varepsilon}\|^2$ 被扩展为 $\|\boldsymbol{D\varepsilon}\|^2$. 其中 $\boldsymbol{D}=\mathrm{diag}(v_1,v_2,\cdots,v_N)$ 表示误差的权值对角阵. RELM 的模型进一步修正为

$$\arg\min_{\boldsymbol{\beta}}\left(\frac{1}{2}\|\boldsymbol{\beta}\|^2+\frac{1}{2}\gamma\|\boldsymbol{D\varepsilon}\|^2\right),$$

$$\text{s.t.}\ \sum_{i=1}^{\widetilde{N}}\boldsymbol{\beta}_i g(\boldsymbol{a}_i\cdot\boldsymbol{x}_j+b_i)-\boldsymbol{t}_j=\boldsymbol{\varepsilon}_j,\ \ j=1,2,\cdots,N.$$

上式是条件极值问题，通过拉格朗日方程转换为无条件极值问题进行求解：

$$\begin{aligned}\ell(\boldsymbol{\beta},\boldsymbol{\varepsilon},\boldsymbol{\alpha})&=\frac{\gamma}{2}\|\boldsymbol{D\varepsilon}\|^2+\frac{1}{2}\|\boldsymbol{\beta}\|^2-\\&\quad\sum_{j=1}^{N}\alpha_j(g(\boldsymbol{a}_i\cdot\boldsymbol{x}_j+b_i)-\boldsymbol{t}_j-\boldsymbol{\varepsilon}_j)\\&=\frac{\gamma}{2}\|\boldsymbol{D\varepsilon}\|^2+\frac{1}{2}\|\boldsymbol{\beta}\|^2-\boldsymbol{\alpha}(\boldsymbol{H\beta}-\boldsymbol{T}-\boldsymbol{\varepsilon})\end{aligned}\tag{4}$$

其中 $\boldsymbol{\alpha}=[\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_N]$;$\boldsymbol{\alpha}_j\in R^m(j=1,2,\cdots,N)$代表拉格朗日乘子.

求拉格朗日方程的梯度并令其为 0:

$$\begin{cases}\dfrac{\partial \ell}{\partial \boldsymbol{\beta}}\rightarrow \boldsymbol{\beta}^{\mathrm{T}}=\boldsymbol{\alpha}\boldsymbol{H} & (5a)\\ \dfrac{\partial \ell}{\partial \boldsymbol{\varepsilon}}\rightarrow \gamma\boldsymbol{\varepsilon}^{\mathrm{T}}\boldsymbol{D}^2+\boldsymbol{\alpha}=0 & (5b)\\ \dfrac{\partial \ell}{\partial \boldsymbol{\alpha}}\rightarrow \boldsymbol{H}\boldsymbol{\beta}-\boldsymbol{T}-\boldsymbol{\varepsilon}=0 & (5c)\end{cases}$$

把方程(5c)代入方程(5b)得

$$\boldsymbol{\alpha}=-\gamma(\boldsymbol{H}\boldsymbol{\beta}-\boldsymbol{T})^{\mathrm{T}}\boldsymbol{D}^2 \tag{6}$$

把式(6)代入方程(5a)得

$$\boldsymbol{\beta}=\left(\frac{\boldsymbol{I}}{\gamma}+\boldsymbol{H}^{\mathrm{T}}\boldsymbol{D}^2\boldsymbol{H}\right)^{\dagger}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{D}^2\boldsymbol{T} \tag{7}$$

表达式(7)只含有一个 $\widetilde{N}\times\widetilde{N}(\widetilde{N}\ll N)$矩阵的逆操作,所以计算 $\boldsymbol{\beta}$ 的速度非常快.

5.1 无权 RELM

在实际应用中,如果数据集中离群点很少,对模型没有太大影响,那么为了加快训练速度,可以认为每个样本的误差权值相同,此时矩阵 $\boldsymbol{D}=\mathrm{diag}(v_1,v_2,\cdots,v_N)$将是一个单位阵,无须计算.我们称这种情况的 RELM 为无权 RELM,无权 RELM 算法可归结如下:

算法 1. 无权 RELM.

给定一个训练集$\aleph=\{(\boldsymbol{x}_i,\boldsymbol{t}_i)\mid \boldsymbol{x}_i\in R^n,\ \boldsymbol{t}_i\in R^m,i=1,2,\cdots,N\}$、激励函数 $g(x)$及隐藏层结点数 $\widetilde{N}$,

(1)随机指定输入权值 $\boldsymbol{a}_i$、偏差 $b_i(i=1,2,\cdots,\widetilde{N})$.

(2)计算隐藏层输出矩阵

$$\boldsymbol{H}=\begin{bmatrix}g(\boldsymbol{a}_1\cdot\boldsymbol{x}_1+b_1) & \cdots & g(\boldsymbol{a}_{\widetilde{N}}\cdot\boldsymbol{x}_1+b_{\widetilde{N}})\\ \vdots & \cdots & \vdots\\ g(\boldsymbol{a}_1\cdot\boldsymbol{x}_N+b_1) & \cdots & g(\boldsymbol{a}_{\widetilde{N}}\cdot\boldsymbol{x}_N+b_{\widetilde{N}})\end{bmatrix}_{N\times\widetilde{N}}.$$

(3)计算输出权值 $\boldsymbol{\beta}$:$\boldsymbol{\beta}=\left(\frac{\boldsymbol{I}}{\gamma}+\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}\right)^{\dagger}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{T}$.

通过观察不难看出,RELM 与 ELM 计算量基本一样.其实 ELM 是未加权 RELM 的一种特殊情况.

定理 3. 当 $\gamma\rightarrow\infty$时,未加权 RELM 将退化为 ELM.

证明. 若 $\gamma\rightarrow\infty$,则$\frac{\boldsymbol{I}}{\gamma}\rightarrow 0$,因此有

$$\boldsymbol{\beta}=\left(\frac{\boldsymbol{I}}{\gamma}+\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}\right)^{\dagger}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{T}=(\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H})^{\dagger}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{T}$$
$$=\boldsymbol{H}^{\dagger}(\boldsymbol{H}^{\mathrm{T}})^{\dagger}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{T}=\boldsymbol{H}^{\dagger}\boldsymbol{T}.$$

证毕.

5.2 加权 RELM

与无权 RELM 相反,如果数据含有离群点,那么使用加权 RELM 有一定的抗干扰能力,这可以从后面"SinC"数据集离群点加入前后的实验对比中看出.加权 RELM 需要计算误差的权值,权值计算已有很多论述[7,14],这里采用文献[15]提到的方法:

$$v_j=\begin{cases}1, & |\boldsymbol{\varepsilon}_j/\hat{s}|\le c_1\\ \dfrac{c_2-|\boldsymbol{\varepsilon}_j/\hat{s}|}{c_2-c_1}, & c_1\le|\boldsymbol{\varepsilon}_j/\hat{s}|\le c_2\\ 10^{-4}, & \text{其它}\end{cases},$$

其中 $\boldsymbol{\varepsilon}_j=-\frac{\boldsymbol{\alpha}_j}{\gamma}$,它是无权 RELM 计算得到的样本误差,$\hat{s}$ 是误差 $\boldsymbol{\varepsilon}_j$ 的标准偏差(standard deviation)估计,可通过公式 $\hat{s}=1.483MAD(x_j)$计算. MAD(Median Absolute Deviation)表示绝对中位差.根据高斯分布可知:基本不存在大于 $2.5\hat{s}$ 的误差,因此常量 c_1 和 c_2 通常被置为 $c_1=2.5$,$c_2=3$[7].综上所述,RELM 算法可归结如下:

算法 2. 加权 RELM.

给定一个训练集$\aleph=\{(\boldsymbol{x}_i,\boldsymbol{t}_i)\mid \boldsymbol{x}_i\in R^n,\ \boldsymbol{t}_i\in R^m,i=1,2,\cdots,N\}$、激励函数 $g(x)$以及隐藏层结点数 $\widetilde{N}$,

(1)随机指定输入权值 $\boldsymbol{a}_i$、偏差 $b_i(i=1,2,\cdots,\widetilde{N})$并且计算隐藏层输出矩阵 $\boldsymbol{H}$.

(2)$\boldsymbol{\beta}=\left(\frac{\boldsymbol{I}}{\gamma}+\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}\right)^{\dagger}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{T}$.

(3)$\boldsymbol{\alpha}=-\gamma(\boldsymbol{H}\boldsymbol{\beta}-\boldsymbol{T})^{\mathrm{T}}$.

(4)$\boldsymbol{\varepsilon}_i=\frac{\boldsymbol{\alpha}_i}{\gamma}(i=1,2,\cdots,N)$.

(5)$\hat{s}=1.483MAD(\boldsymbol{x}_j)$.

(6)$\boldsymbol{D}=\mathrm{diag}(v_1,v_2,\cdots,v_N)$:

$$v_j=\begin{cases}1, & |\boldsymbol{\varepsilon}_j/\hat{s}|\le c_1\\ \dfrac{c_2-|\boldsymbol{\varepsilon}_j/\hat{s}|}{c_2-c_1}, & c_1\le|\boldsymbol{\varepsilon}_j/\hat{s}|\le c_2\\ 10^{-4}, & \text{其它}\end{cases}.$$

(7)$\boldsymbol{\beta}=\left(\frac{\boldsymbol{I}}{\gamma}+\boldsymbol{H}^{\mathrm{T}}\boldsymbol{D}^2\boldsymbol{H}\right)^{\dagger}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{D}^2\boldsymbol{T}$.

加权 RELM 多了计算权值的过程,时间消耗有所延长,因此如果实际应用中对训练时间要求很强,那么用无权 RELM 比较合适.在下面的实验中,除为了验证 RELM 的鲁棒性在"SinC"数据集上采用加权 RELM 和 ELM 进行比较外,其它数据集的实验一律采用无权 RELM 和 ELM 进行比较.

RELM 与 ELM 相比,具有如下特点:

(1)方程组的解是 $\boldsymbol{H}\boldsymbol{\beta}=\boldsymbol{T}$ 的一个加权最小二乘解:

$$\|\boldsymbol{H}\hat{\boldsymbol{\beta}}-\boldsymbol{T}\|=\|\boldsymbol{H}(\boldsymbol{H}^{\mathrm{T}}\boldsymbol{D}^2\boldsymbol{H})^{\dagger}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{D}^2\boldsymbol{T}-\boldsymbol{T}\|$$
$$=\arg\min_{\boldsymbol{\beta}}\|\boldsymbol{H}\boldsymbol{\beta}-\boldsymbol{T}\|.$$

这个解不但可以达到最小的训练误差,同时对离群点具有一定的抗干扰能力.

(2)通过引入调节参数 γ,代价函数不仅包括经

验风险，还包括结构风险，这使得方程组的解不仅获得尽可能小的训练误差，而且能使边缘距离最大化，从而具有更好的泛化性能：

$$\left\| \boldsymbol{H}\left(\frac{\boldsymbol{I}}{\gamma}+\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}\right)^{\dagger}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{T}-\boldsymbol{T}\right\| = \arg\min_{\boldsymbol{\beta}}(\gamma\|\boldsymbol{H}\boldsymbol{\beta}-\boldsymbol{T}\|+\|\boldsymbol{\beta}\|^{2}).$$

6　性能评估

这里我们通过实验结果比较 RELM、ELM、BP 和支持向量机（Support Vector Machine，SVM）[12-13]的性能. RELM、ELM 和BP的执行环境是 Matlab7.0，SVM 的执行环境是 C 语言. RELM 由我们自己实现，ELM 的源代码可以从 Huang 的个人主页直接下载①，而 BP 算法已经集成在 Matlab 自带的神经网络工具箱中，可以直接使用. BP 算法有很多变种，我们选择最快的 Levenberg-Marquardt算法来进行实验. SVM 算法我们采用 C 语言实现的 SVM 包：LibSVM②. RELM、ELM 和 BP 的激励函数都选择“Sigmoid”函数：$g(x)=1/(1+\exp(-x))$，而 SVM 的核函数选择径向基函数. 实验数据的输入一律归一化到[0,1]范围内，而输出则归一化到[−1,1]范围内.

值得指出的是，这里汇总的实验结果都是每种算法能够达到的最优实验结果. 对于 SVM，我们采用Hsu和Lin提出的排列组合方式[16]选择最优的参数 γ 和 C：$\gamma=[2^{4},2^{3},\cdots,2^{-10}]$，$C=[2^{12},2^{11},\cdots,2^{-2}]$. 共有 $15\times15=225$ 种组合，对每一种组合(γ,C)，进行 50 次随机实验，并对最佳平均值进行汇总. 对于 RELM，我们采用类似于 SVM 的方式选择最优的参数 γ 和隐藏层结点数 $\widetilde{N}$：$\gamma=[2^{-50},2^{-49},\cdots,2^{50}]$，$\widetilde{N}=[5,10,\cdots,\widetilde{N}_{\max}]$（$\widetilde{N}_{\max}$根据具体数据集设定）. 对于所产生的每个组合$(\gamma,\widetilde{N})$，进行 50 次随机实验，并对最佳平均值进行汇总. 对于 ELM 和 BP，隐藏层结点的个数初始取 5，每次递增 5，并基于 5-折交叉验证的方法选择最优（接近）的数目，然后进行 50 次实验并将最佳平均结果进行汇总.

6.1　回归问题

6.1.1　人工数据：“SinC”

“SinC”函数表达式：$y(x)=\begin{cases}\sin x/x, & x\neq0\\ 1, & x=0\end{cases}$.

数据产生方法：在区间（−10,10）内随机产生 5000 个训练样本和测试样本，并在所有训练样本上附加取值范围为[−0.2,0.2]的随机噪声，而测试数据无噪声. 各种算法的性能见表 1. 从表 1 可以看出 RELM 的 RMSE（Root Mean Square Error，均方根误差）比 ELM 小，分别为 **0.0078** 和 0.0097；不过 RELM 训练时间比 ELM 稍长；RELM 的 RMSE 明显比 BP 算法和 SVM 算法要小，而训练时间却比 BP 和 SVM 缩短了上百倍. 由此可见在“SinC”数据集上，RELM 综合性能最好.

表 1　4 种算法在“SinC”数据集上的性能比较

	时间/s		RMSE		Dev		支持向量个数/隐藏层结点数
	Training	Testing	Training	Testing	Training	Testing	
RELM	0.106	0.024	0.1154	**0.0078**	0.00077	**0.0010**	20
ELM	0.096	0.024	0.1148	0.0097	0.0037	0.0028	20
BP	16.354	0.035	0.1196	0.0159	0.0042	0.0041	20
SVM	979.538	4.545	0.1149	0.0130	0.0007	0.0012	2500

为了比较 RELM 和 ELM 算法的鲁棒性，“SinC”训练集中加入了一些离群点后进行重新实验. 实验结果见图 1，从图中可以看出 ELM 的预测曲线明显脱离实际曲线，说明其受到离群点的干扰很大. 而 RELM 的预测曲线仍能完好地拟合实际曲线，说明 RELM 具有一定的抗干扰能力.

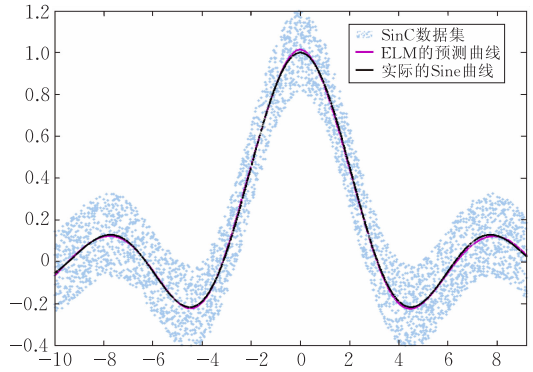
6.1.2　实际回归问题

我们在 13 种真实数据集③上将 RELM 与 ELM、BP、SVM 进行比较，数据集信息见表 2. 4 种算法的 RMSE 见表 3. 从表 3 可以看出，RELM 在大多数据集上的测试 RMSE 比 ELM、BP、SVM 小，说明其有更好的泛化性能（如果两种算法的 RMSE 相差大于 0.005 时，较好的 RMSE 加粗表示）；表 4 汇总了 4 种算法的时间消耗，从表 4 可以看出 RELM 的训练速度和 ELM 相差无几，却比 BP 和 SVM 快很多倍. 但是由于 BP 具有最紧凑网络结构（隐藏层结点数最少），在 4 种算法中 BP 测试时间最短；表 5 汇总了 4 种算法的标准偏差.

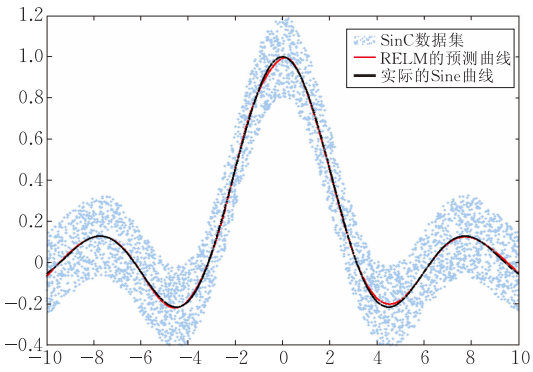
① ELM Source Codes：http://www.ntu.edu.sg/home/egbhuang/

② SVM Source Codes：http://www.csie.ntu.edu.tw/~cjlin/libsvm/

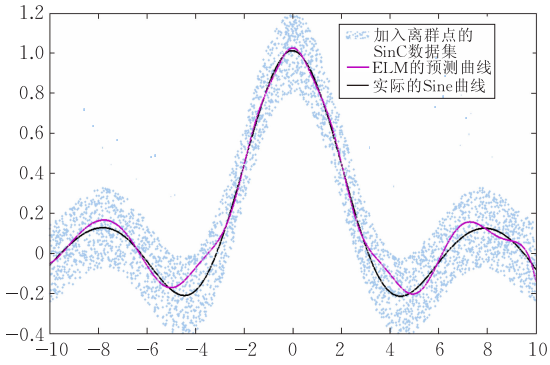
③ http://www.niaad.liacc.up.pt/~ltorgo/Regression/ds_menu.html



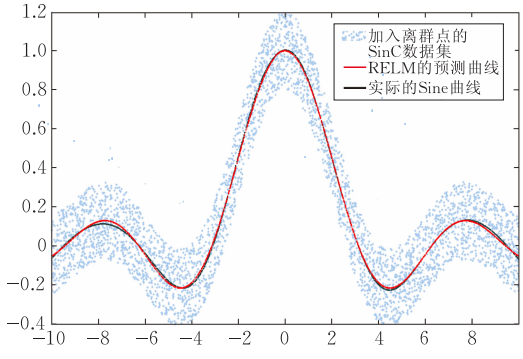
(a) ELM拟合曲线(无离群点)



(b) RELM拟合曲线(无离群点)



(c) ELM拟合曲线(有离群点)



(d) RELM拟合曲线(有离群点)

图 1 加入离群点前后“SinC”曲线拟合效果比较

表 2 回归数据集信息

数据集	样本数目		属性数目		数据集	样本数目		属性数目	
	训练样本	测试样本	连续属性	标称属性		训练样本	测试样本	连续属性	标称属性
Abalone	2000	2177	7	1	Machine CPU	100	109	6	0
Delta ailerons	3000	4129	6	0	Servo	80	87	0	4
Delta elevators	4000	5517	6	0	Breast cancer	100	94	32	0
Computer activity	4000	4192	8	0	Bank domains	4500	3692	8	0
Census	10000	12784	8	0	California housing	8000	1246	8	0
Auto price	80	79	14	1	Stocks domain	450	500	10	0
Triazines	100	86	60	0					

表 3 4 种不同算法的均方差(RMSE)比较

数据集	BP 的 RMSE		SVM 的 RMSE		ELM 的 RMSE		RELM 的 RMSE	
	训练样本	测试样本	训练样本	测试样本	训练样本	测试样本	训练样本	测试样本
Abalone	0.0785	0.0874	0.0759	0.0784	0.0803	0.0824	0.0746	**0.0764**
Delta ailerons	0.0409	0.0481	0.0418	0.0429	0.0423	0.0431	0.0280	**0.0388**
Delta elevators	0.0544	0.0592	0.0534	0.0540	0.0550	0.0568	0.0017	**0.0179**
Computer activity	0.0464	0.0470	0.0464	0.0470	0.0316	0.0382	0.0321	**0.0185**
Census(house8L)	0.0718	0.0746	0.0718	0.0746	0.0624	0.0660	0.0624	**0.0360**
Auto price	0.0652	0.0937	0.0652	0.0937	0.0754	0.0994	0.0724	**0.0753**
Triazines	0.1432	0.1829	0.1432	0.1829	0.1897	0.2002	0.1877	**0.1801**
Machine CPU	0.0352	0.0826	0.0574	0.0811	0.0332	0.0539	0.0168	**0.0220**
Servo	0.0840	0.1177	0.0840	0.1177	0.0707	0.1196	0.0721	**0.089**
Breast cancer	0.2278	0.2643	0.2278	0.2643	0.2470	0.2679	0.2145	**0.201**
Bank domains	0.0454	0.0467	0.0454	0.0467	0.0406	0.036	0.041	**0.015**
California housing	0.1089	0.1180	0.1089	0.1180	0.1217	0.1267	0.1213	**0.1170**
Stocks domain	0.0503	0.0518	0.0503	0.0518	0.0251	0.0348	0.022	**0.0132**

表 4　4 种不同算法的时间比较　　(单位：s)

数据集	BP 的时间		SVM 的时间		ELM 的时间		RELM 的时间	
	训练样本	测试样本	训练样本	测试样本	训练样本	测试样本	训练样本	测试样本
Abalone	1.330	0.005	1.221	0.244	0.009	0.023	0.010	0.023
Delta ailerons	2.085	0.012	0.510	0.169	0.045	0.048	0.046	0.048
Delta elevators	0.904	0.009	0.849	0.445	0.213	0.155	0.221	0.155
Computer activity	51.091	0.052	0.769	0.229	0.224	0.130	0.297	0.132
Census(house8L)	6.110	0.035	8.523	3.142	0.818	0.477	0.827	0.484
Auto price	0.186	<10e-4	0.003	0.035	0.001	<10e-4	0.049	<10e-4
Triazines	0.108	<10e-4	0.007	0.041	<10e-4	<10e-4	<10e-4	<10e-4
Machine CPU	0.178	<10e-4	0.001	0.032	0.001	<10e-4	0.001	<10e-4
Servo	0.185	<10e-4	0.003	0.030	<10e-4	<10e-4	<10e-4	<10e-4
Breast cancer	0.292	<10e-4	0.005	0.045	<10e-4	<10e-4	<10e-4	<10e-4
Bank domains	5.686	0.035	1.218	0.229	0.487	0.167	0.548	0.169
California housing	4.948	0.036	56.200	7.919	0.847	0.230	0.940	0.234
Stocks domain	0.794	0.005	0.052	0.048	0.013	0.023	0.014	0.023

表 5　4 种不同算法的标准偏差(standard deviations)比较

数据集	BP 的标准偏差		SVM 的标准偏差		ELM的标准偏差		RELM 的标准偏差	
	训练样本	测试样本	训练样本	测试样本	训练样本	测试样本	训练样本	测试样本
Abalone	0.0011	0.0034	0.0015	0.0013	0.0049	0.0058	0.0051	0.0056
Delta ailerons	0.0015	0.0015	0.0012	0.0010	0.0030	0.0035	0.0028	0.0034
Delta elevators	0.0007	0.0003	0.0006	0.0005	0.0028	0.0029	0.0025	0.0029
Computer activity	0.0007	0.0007	0.0015	0.0016	0.0005	0.0033	0.0004	0.0034
Census(house8L)	0.0011	0.0050	0.0013	0.0013	0.001	0.0017	0.002	0.0026
Auto price	0.0405	0.0231	0.0025	0.0040	0.0119	0.0119	0.0123	0.0139
Triazines	0.0656	0.0531	0.0152	0.0163	0.0212	0.0209	0.0213	0.0207
Machine CPU	0.0192	0.0715	0.0083	0.0180	0.0060	0.0156	0.0056	0.0147
Servo	0.0313	0.0475	0.0406	0.0185	0.0121	0.0113	0.0119	0.0111
Breast cancer	0.1529	0.0962	0.0115	0.0151	0.0121	0.0167	0.0127	0.0167
Bank domains	0.0006	0.0004	0.0005	0.0008	0.0006	0.0009	0.0006	0.0008
California housing	0.0045	0.0026	0.0012	0.0011	0.0021	0.0033	0.0020	0.0026
Stocks domain	0.0012	0.0022	0.0016	0.0022	0.0011	0.0016	0.0014	0.0015

前面提到当 $\gamma \to \infty$ 时，RELM 将退化为 ELM. 为了说明这一点，我们以数据集“Triazines”为例展示 RELM 的性能(RMSE)随 γ 变化情况. 如图 2 所示，可以看出 RELM 的性能首先随着 γ 的增大不断提高(越小越好)，当 $\gamma = 2^{-2}$ 时，RELM 的性能达到最好，比 ELM 提高了 0.05. 之后，随着 γ 的增大，RELM 的性能不断降低，并逐渐与 ELM 的性能曲线重叠在一起，这说明当 $\gamma \to \infty$ 时，RELM 退化为 ELM，由此可见 RELM 的精度至少能与 ELM 相当.

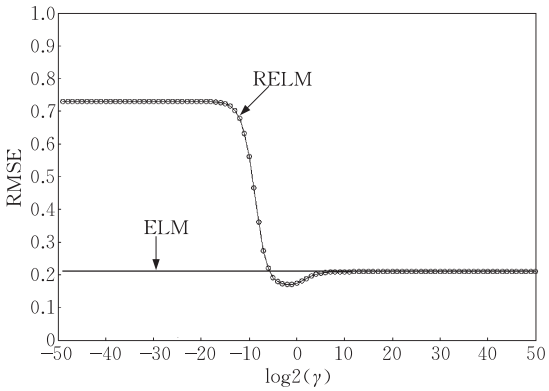


图 2　RELM 的性能随 γ 变化的曲线

6.2　分　类

在这一部分，我们来给出 RELM 在分类中的性能测试. 数据集信息见表 6，包括 Banana 数据集[①]以及一些来自 UCI 的数据集[②]. Banana 数据集中冗余数据已被删除，只选用其中的 5200 个不同的训练样本. 由于 Forest-type 数据集规模很大，基于梯度的 BP 算法在传统 PC 上会产生内存溢出，因此我们直接引用文献[17]中的实验结果. SVM 的参数设置是 $C=10$. 不同算法的性能见表 7，从表 7 可以看出 RELM 的运行速度和 ELM 几乎一样快，并获得更好的泛化性能.

① http://www.?rst.gmd.de/~raetsch/data
② http://www.ics.uci.edu/mlearn/MLRepository.html

表 6 分类问题数据集的信息

数据集	训练样本	测试样本	属性	类别	数据集	训练样本	测试样本	属性	类别
Diabetes	576	192	8	2	Shuttle	43500	14500	9	7
Satimage	4400	2000	36	7	Banana	5200	490000	2	2
Segmen	1500	810	18	7	Forest-type	100000	481012	54	7

表 7 4 种算法在分类问题上的性能比较

数据集	算法	时间/s		分类成功率/%		Dev/%		支持向量个数/隐藏层结点数
		训练样本	测试样本	训练样本	测试样本	训练样本	测试样本	
Diabetes	RELM	0.011	0.002	78.69	**78.19**	1.090	2.55	20
	ELM	0.009	0.002	78.68	77.57	1.180	2.85	20
	BP	2.246	0.003	86.63	74.73	1.700	3.2	20
	SVM	0.139	0.054	78.76	77.31	0.910	2.35	315
Satimage	RELM	11.186	0.210	92.47	**89.41**	0.030	0.0056	500
	ELM	11.126	0.274	93.52	89.04	1.460	1.50	500
	BP	9366.8	0.064	95.26	82.34	0.970	1.25	100
Segment	RELM	1.059	0.057	98.13	**95.05**	0.160	0.0074	200
	ELM	1.044	0.056	97.35	94.95	0.320	0.78	200
	BP	3538.9	0.032	96.92	86.27	0.450	1.80	100
Shuttle	RELM	4.293	0.193	99.55	99.40	0.0072	0.00655	50
	ELM	4.280	0.185	99.65	99.40	0.120	0.12	50
	BP	4572.8	0.177	99.77	99.27	0.100	0.13	50
Banana	RELM	1.648	16.180	92.33	**91.78**	0.097	0.105	100
	ELM	1.633	16.164	92.36	91.57	0.170	0.25	100
	BP	4572.8	17.002	90.26	88.09	0.270	0.70	100
Forest-type	RELM	1.361	0.581	92.25	**90.33**	0.012	0.013	200
	ELM	1.204	0.580	92.35	90.21	0.026	0.024	200
	BP[16]	12min	N/A	82.44	81.85	N/A	N/A	100
	SVM	517.2	280.244	91.70	89.90	N/A	N/A	31805

7 结束语

RELM 打破了传统 BP 算法的参数迭代调整的思想,从而获得了快速学习的能力,从不同数据集的实验可以看出 RELM 比 BP、SVM 速度提高很多倍(通常是 10 倍以上),而泛化性能却比 BP、SVM 高,这无疑为神经网络应用到实时环境提供了有效途径.与 ELM 相比,RELM 不但继承了 ELM 快速训练的特点,还通过引入参数 γ 使得模型可以根据数据集的特点进行微调,从而得到更好的泛化性能,增强了系统的可控性.另外如果数据集中存在明显的噪声,那么可以使用加权 RELM,达到降噪的目的,从对"SinC"数据集的实验可以看出加权 RELM 对噪声有一定的抗干扰能力.在我们下一步研究中,我们打算将在线学习与 RELM 相结合,提出一种能够处理流数据的 RELM;将 SOM[18]、主动学习等技术与 RELM 相结合以期得到更好的性能.在应用方面,我们打算将 RELM 应用到文本分类、协作过滤、数字水印等领域,尤其是文本分类与协作过滤都存在高维稀疏问题,传统方法非常耗时,如何研究一种有效的快速 Sparse-RELM 将是另一个值得研究的问题.

参 考 文 献

[1] Hornik K. Approximation capabilities of multilayer feedforward networks. Neural Networks, 1991, 4(2): 251-257

[2] Leshno M, Lin V Y, Pinkus A, Schocken S. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. Neural Networks, 1993, 6(6): 861-867

[3] Huang G-B, Babri H A. Upper bounds on the number of hidden neurons in feedforward networks with arbitrary bounded nonlinear activation functions. IEEE Transactions on Neural Networks, 1998, 9(1): 224-229

[4] Huang G-B. Learning capability and storage capacity of two hidden-layer feedforward networks. IEEE Transactions on Neural Networks, 2003, 14(2): 274-281

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

[6] Vapnik V N. The Nature of Statistical Learning Theory. New York: Springer, 1995

[7] Rousseeuw P J, Leroy A. Robust Regression and Outlier Detection. New York: Wiley, 1987

[8] Rumelhart D E, McClelland J L. Parallel Distributed Processing. Cambridge: MIT Press, 1986, 1(2): 125-187

[9] Cristianini N, Shawe-Taylor J. An Introduction to Support Vector Machines. Cambridge: Cambridge University Press, 2000

[10] Tamura S, Tateishi M. Capabilities of a four-layered feedforward neural network: Four layers versus three. IEEE Transactions on Neural Networks, 1997, 8(2): 251-255

[11] Haykin S. Neural Networks: A Comprehensive Foundation. New Jersey: Prentice Hall, 1999

[12] Deng Nai-Yang, Tian Ying-Jie. A New Method of Data Mining—Support Vector Machine. Beijing: Science Press, 2004 (in Chinese)
(邓乃扬，田英杰. 数据挖掘中的新方法——支持向量机. 北京：科学出版社，2004)

[13] Zhang Xue-Gong. Introduction to statistical learning theory and support vector machines. Acta Automatica Sinica, 2000, 26(1): 32-42(in Chinese)
(张学工. 关于统计学习理论与支持向量机. 自动化学报，2000，26(1)：32-42)

[14] David H A. Early sample measures of variability. Statistical Science, 1998, 13(4): 368-377

[15] Suykens J A K, De Brabanter J, Lukas L, Vandewaile J. Weighted least squares support vector machines: Robustness and sparse approximation. Neurocomputing, 2002, 48(1): 85-105

[16] Hsu C-W, Lin C-J. A comparison of methods for multiclass support vector machines. IEEE Transactions on Neural Networks, 2002, 13(2): 415-425

[17] Collobert R, Bengio S, Bengio Y. A parallel mixtures of SVMs for very large scale problems. Neural Computation, 2002, 14(5): 1105-1114

[18] Keerthi S S, Shevade S K. SMO algorithm for least squares SVM formulations. Neural Computation, 2003, 15(2): 487-507

DENG Wan-Yu, born in 1979, Ph.D. candidate, lecturer. His research interests include machine learning, collaborative filtering and personalized service.

ZHENG Qing-Hua, born in 1969, Ph. D., professor, Ph. D. supervisor. His research interests include intelligent learning theory, network security.

CHEN Lin, born in 1977, M. S., lecturer. Her research interests include machine learning, collaborative filtering and personalized service.

XU Xue-Bin, born in 1974, M. S., engineer. His research interests include machine learning, graph processing.

Background

Extreme Learning Machine proposed recently has attracted a lot of attention for fast training speed. Based on structural risk minimization principle and weighted least square, the authors presents a novel algorithm called Regularized Extreme Learning Machine. The generalization performance is improved significantly in most case without increasing training consuming comparing with the original Extreme Learning Machine algorithm. The proposed RELM can overcome overfitting of ELM, and achieve more robust than ELM. The algorithm has wide application potential. The authors have used RELM to personalized resource recommendation.