

基于极速学习的粗糙 RBF 神经网络

马 刚^{1,2}, 丁世飞^{1,2}, 史忠植²

(1 中国矿业大学 计算机科学与技术学院, 江苏 徐州 221008; 2 中国科学院 计算技术研究所, 北京 100190)

摘 要: 提出了一种用于训练粗糙 RBF 神经网络(rough RBF neural networks, R-RBF)的极速学习机(extreme learning machine, ELM)方法, 通过引入矩阵的 Moore-Penrose 逆, 将传统的迭代学习方法转换为一种求线性方程的极小范数最小二乘解的方法. 实验证明, 在训练精度、训练时间上都能够达到非常优越的性能, 其泛化精度能够提升 50% 以上.

关键词: ELM; R-RBF; Moore-Penrose; 极小范数最小二乘解

中图分类号: TP393

文献标识码: A

文章编号: 1000-7180(2012)08-0009-06

Rough RBF Neural Network Based on Extreme Learning

MA Gang^{1,2}, DING Shi-fei^{1,2}, SHI Zhong-zhi²

(1 School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221008, China;

2 Institute of Computing Technology, Chinese Academy of Science, Beijing 100190, China)

Abstract: The paper proposes a method of training rough RBF neural networks(R-RBF) using the extreme learning machine(ELM), which converts the traditional iterative training method to solve norm least-squares solution of general linear system by introducing Moore-Penrose inverse. Experiments show that it can reach a very superior performance in both time and accuracy when ELM trains the Rough RBF Neural Networks, which can improve the generalization accuracy more than 50% compared with the traditional thinking of adjusting parameters iteratively.

Key words: ELM; R-RBF; Moore-Penrose; norm least-squares solution

1 引言

粗糙 RBF 神经网络(rough RBF neural networks, R-RBF)是基于传统 RBF 神经网络为了应对处理信息冗余而提出的. 因此, 粗糙 RBF 神经网络除了具有神经网络的已有特性外, 还有着非常强的信息冗余处理能力^[1]. 但是, 粗糙 RBF 神经网络和传统神经网络一样, 都面临着学习效率和泛化精度的问题. 神经网络的传统学习方法都是基于参数迭代调整思想的, 而参数迭代调整最大的一个缺点就是耗时较多, 严重影响了网络的学习时间效率. G.-B. Huang 在总结参数迭代调整这一学习思想时, 提出了一种称作为极速学习机(extreme learn-

ing machine, ELM)来训练单隐层前向神经网络的思想^[2]. 本文就是将 ELM 的思想引入到粗糙 RBF 神经网络, 以达到网络的整个学习过程无需迭代, 且一次性完成, 从而解决了粗糙 RBF 神经网络利用传统参数迭代思想进行学习的耗时问题, 极大提高了整个网络的学习效率.

通过实验分析验证可知, 利用极速学习机来进行粗糙 RBF 神经网络的学习, 能够解决基于参数迭代思想的传统神经网络学习方法关于时间的效率问题, 而且与传统的迭代思想相比, 还具有更高的学习泛化精度. 因此, 本文提出的方法确实有效可行.

2 粗糙集样本属性重要度的确定

2.1 粗糙集相关概念介绍

定义 1 信息系统是系统 $S=(U,A)$, 其中 $U=\{u_1, u_2, \dots, u_n\}$ 是有限非空集, 称为论域或对象空间, U 中的元素称为对象或实例; $A=\{a_1, a_2, \dots, a_{|A|}\}$ 也是有限非空集, A 中的元素称为属性, 对于每个 $a \in A$, 有一个映射 $f: U \rightarrow a(U)$, 且 $a(U)=\{a(u) | u \in U\}$, 其中 $a(U)$ 称为属性 a 的值域 V . 论域 U 、属性集 A 、值域 V 和映射 f 组成了信息系统的四个要素, 因此信息系统 S 也可以表示为 $S=(U, A, V, f)^{[3]}$.

如果 $A=C \cup D$, $C \cap D = \emptyset$, 则称信息系统 (U, A) 为一个决策表, 其中 C 中的属性称为条件属性, D 中的属性称为决策属性.

定义 2 不可区分关系是信息系统 $S=(U, C \cup D)$ 上的二元等价关系 $IND(P)$, 其中 P 为 $C \cup D$ 的任意子集, $P \neq \emptyset$, $IND(P)=\{\forall a \in P | (x, y) \in U \times U; a(x)=a(y)\}^{[4]}$.

由 $IND(P)$ 导出的所有等价类的集合记为 $U/IND(P)$, 它构成了 U 的一个划分, 称为 U 的一个知识. $U/IND(P)$ 可以简记为 U/P .

定义 3 设 R 为论域 U 上的等价关系, 则称 (U, R) 为近似空间.

定义 4 设 (U, R) 为近似空间, 令 $U/R=\{X_1, X_2, \dots, X_n\}$. 记 $P(X_i)=|X_i|/|U|$, 则知识 U/R 的熵定义为^[5]

$$H\left(\frac{U}{R}\right)=-\sum_{i=1}^n P(X_i) \log P(X_i) \quad (1)$$

定理 1 设 $S=(U, A)$ 是一个信息系统, $P, Q \subseteq A$, 若 $IND(Q) \subseteq IND(P)$, 则 $H(U/P) \leq H(U/Q)^{[6]}$.

定理 1 表明知识熵随着信息粒度的变小(通过更细的划分)而单调增加.

2.2 样本属性知识熵的度量

式(1)所给出的知识熵的定义及计算公式是从信息论的角度引入的, 因为近似空间中没有定义概率测度, 可以将论域上的任何一个等价关系形成的划分看成是定义在论域上的子集所组成的 δ 代数上的一个随机变量, 此时每一个集合上的概率定义为该集合的基数与论域的基数之比, 即 $P(X_i)=|X_i|/|U|$.

根据上述方法所计算出的熵, 其取值范围为 $[1, +\infty)$, 显然不符合权值的要求. 于是由文献[3]可以

得到知识熵的另一种定义方式:

定义 5 设 $\Omega=\{X_1, X_2, \dots, X_n\}$ 是 U 上的划分, 记 $P(X_i)=|X_i|/|U|$, 则 Ω 的知识熵定义为

$$H(\Omega)=\sum_{i=1}^n P(X_i) \times P(X_i^c) \quad (2)$$

在文献[3]中称为信息量, 在此将其确定为知识熵的另外一种度量方式, $H(\Omega)$ 有如下性质:

- (1) $0 \leq H(\Omega) \leq 1 - \frac{1}{|\Omega|}$;
- (2) $H(\Omega)=0$, 当且仅当 $\Omega=\{U\}$;
- (3) $H(\Omega)=1 - \frac{1}{|\Omega|}$, 当且仅当 $|X_i|=|X_j|$, $(i, j=1, 2, \dots, n)$;
- (4) $P(X_k)=1 - P(X_i)$.

上述知识熵的度量方法在计算上较为简便, 由性质(1)可知: 如果 $n=1/|U|$, 则 $0 \leq H(\Omega) \leq 1 - \frac{1}{n}$, 这样更符合权值的取值要求; 在性质(2)中可以看出: 如果没有不确定性, 则权重(知识熵)即为 0; 性质(3)表明: 权值最大(不确定性最大)时是在分类均匀的情况下, 此时作为消除不确定性的知识就更加重要了.

2.3 样本属性重要度的计算

由于是根据样本属性对样本库的划分能力来确定各特征的权值而不需要任何其他的先验信息, 故该方法具有很强的客观性. 因此, 基于知识熵的样本属性重要度计算步骤如下:

(1) 样本库数据的预处理. 为适应粗糙集方法的处理要求, 需要对连续属性值进行离散化处理, 即选择适当的划分点将整个连续属性的值域范围划分成一些离散区间, 并且用不同的整数值代表每个子区间的属性值.

(2) 计算各样本属性的知识熵. 在样本属性值离散化的基础上, 确定当前属性对样本库的划分, 属性 a_j 的划分为

$$U/IND(a_j)=\left\{\bigcup_{a_j(x)=v_k} x | x \in U, v_k \in v\right\}=(X_1, X_2, \dots, X_n) \quad (3)$$

再根据定义 5 计算其知识熵:

$$\begin{aligned} H(a_j) &= \sum_{i=1}^n P(X_i) \times P(X_i^c) = \\ &= \sum_{i=1}^n P(X_i) \times [1 - P(X_i)] = \\ &= 1 - \sum_{i=1}^n P^2(X_i) \end{aligned} \quad (4)$$

(3) 对重要度进行分配. 根据各个样本属性的知

识熵大小来确定重要度分配,即对知识熵进行归一化处理.重要度计算公式如下:

$$w_j = H(a_j) / \sum H(a_j) \quad (5)$$

3 粗糙 RBF 神经网络极速学习的线性求解数学模型

3.1 粗糙 RBF 神经网络一般线性系统模型及其求解

设一个样本容量为 N , 样本属性个数为 n 的样本集合 $X = (X_1, X_2, \dots, X_N)$, 其中 $X_i = (a_{i1}, a_{i2}, \dots, a_{in})$, 利用式(5)计算得到属性重要度数组 $W = [w_1, w_2, \dots, w_n]$, 然后利用该属性重要度数组 W 对输入的每个样本属性进行相应的属性约简, 产生约简后的样本属性输出 $OutPutX = (OutPutX_1, OutPutX_2, \dots, OutPutX_N)$, 其中 $OutPutX_i = (output_x_{i1}, output_x_{i2}, \dots, output_x_{in})$ 且 $output_x_{ij} = a_{ij} \times w_j, w_j \in W, a_{ij} \in X_i, i$ 表示第 i 个样本, j 表示第 j 个属性. 利用属性重要度进行粗糙集属性约简的 RBF(Rough Set RBF, RS-RBF)神经网络线性结构如图 1 所示.

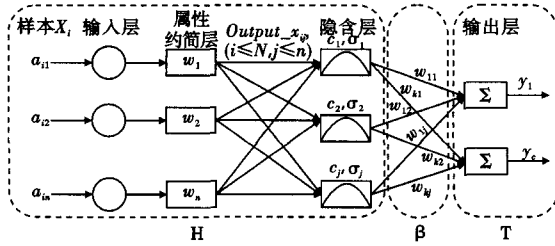


图 1 R-RBF 神经网络的一般线性结构

在图 1 中, $a_{i1}, a_{i2}, \dots, a_{in}$ 表示输入样本 X_i 的条件属性, $y_1, y_2, \dots, y_c$ 表示网络的输出类别, 隐含层的每个神经元对应一个决策模式, 激励函数采用高斯径向基函数:

$$R_k(OutPutX_i) = \exp(-\|OutPutX_i - C_k\|^2 / 2\sigma_k^2) \\ k=1, 2, \dots, j; \quad i=1, 2, \dots, N;$$

$$OutPutX_i = (output_x_{i1}, output_x_{i2}, \dots, output_x_{in}) \quad (6)$$

式中, j 为隐含层神经元的数目; c 为输出层神经元个数; $OutPutX_i$ 为第 i 个样本 $X_i = (a_{i1}, a_{i2}, \dots, a_{in})$ 在属性约简层利用样本属性重要度进行属性约简后得到的属性约简层输出, 也即隐含层的 n 维输入向量; C_k 与 σ_k 分别表示隐含层第 k 个神经元径向基函数的中心和宽度; $\|OutPutX_i - C_k\|$ 表示隐含层输入向量 $OutPutX_i$ 与径向基函数中心 C_k 之间的距离.

最终, 粗糙 RBF 神经网络的输出为

$$y_m = \sum_{k=1}^j w_{mk} R_k(OutPutX_i);$$

$$k=1, 2, \dots, j; m=1, 2, \dots, c; i=1, 2, \dots, N \quad (7)$$

式中, w_{mk} 为隐层第 k 个节点与输出层第 m 个节点之间的连接权值, R_k 是第 k 个隐层节点的输出值, y_m 是输出层第 m 个节点的输出值.

RBF 神经网络是单隐层前馈神经网络(Single-hidden Layer Feedforward Neural Network, SLFN)的一种, 而在本文中所设计的粗糙 RBF 神经网络在实质上也是一种单隐层前馈神经网络, 所以在粗糙 RBF 神经网络的隐层中心和宽度随机给定的情况下, 也可以将等式(7)表示为 $H\beta = T$ (H 为隐层输出矩阵, β 为隐层与输出层之间的连接权值, T 为神经网络的输出值矩阵)这样一种矩阵型的紧凑格式. 显然, 在 H 和 T 已知的情况下, 通过方程 $H\beta = T$ 是非常容易地计算出 β 的值, 也即与 RBF 神经网络其他参数(C_k 和 σ_k)相匹配的输出权值 β .

但是, 在实际的应用中, 对于方程 $H\beta = T$ 的求解, 考虑到矩阵 H 有可能是奇异方阵或是任意的 $m \times n$ 矩阵(一般情况下 $m \neq n$)时, 方程的求解就不能简单的利用 $\beta = H^{-1}T$ 来求解, 而是必须借助于另外的一种方法——求 H 的 Moore-Penrose 逆 H^+ 来求解^[7], 即方程此时的解应该为 $\beta = H^+T$. 在欧几里得空间中, 这与寻求一般线性系统 $Ax = y$ 的极小范数最小二乘解有着相同的过程^[8]. 因此, 可以利用求解一般线性系统 $Ax = y$ 的极小范数最小二乘解的方法来求解方程 $H\beta = T$ 的解 $\hat{\beta}$.

针对于等式(7), 将其改写为一种矩阵型的紧凑表示格式:

$$H\beta = T \quad (8)$$

其中:

$$H(OutPutX, C, \sigma) = H(OutPutX_1, \dots, OutPutX_N, C_1, \dots, C_j, \sigma_1, \dots, \sigma_j) = \\ \begin{bmatrix} R(OutPutX_1, C_1, \sigma_1) & \dots & R(OutPutX_1, C_j, \sigma_j) \\ \vdots & \dots & \vdots \\ R(OutPutX_N, C_1, \sigma_1) & \dots & R(OutPutX_N, C_j, \sigma_j) \end{bmatrix}_{N \times j} \quad (9)$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_j^T \end{bmatrix}_{j \times c}; \quad T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N \times c} \quad (10)$$

在式(9)中, 将 $H(OutPutX, C, \sigma)$ 称作粗糙 RBF 神经网络的隐层输出矩阵, $H(OutPutX, C, \sigma)$ 中的第 k ($k=1, 2, \dots, j$) 列是第 k 个隐层中心节点

所对应的 N 个属性约简样本 ($OutPutX_1, OutPutX_2, \dots, OutPutX_N$) 的输出^[9].

由于粗糙 RBF 神经网络隐层中心参数 C 和 σ 可以随机确定,且无需经过训练,所以针对随机给定的一组隐层中心参数 C_k 和 σ_k (其中 $k=1, 2, \dots, j$) 而言,训练整个粗糙 RBF 神经网络就很容易地转变为了求解线性系统 $H\beta=T$ 的最小二乘解^[10],如下:

$$\|H(C_1, \dots, C_j, \sigma_1, \dots, \sigma_j)\hat{\beta}-T\| = \min_{\beta} \|H(C_1, \dots, C_j, \sigma_1, \dots, \sigma_j)\beta-T\| \quad (11)$$

但是,在大多数的实际应用中 $j \neq N$ (网络的隐层神经元个数不等于训练样本容量,有可能是隐层神经元个数远小于样本容量,也即 $j \ll N$),也即 H 为非方阵或是奇异方阵,此时并没有一个精确的解 β 使得 $H\beta=T$ 成立,所以这就必须借助于前面 3.1 节节所叙述的 Moore-Penrose 广义逆和极小范数最小二乘解方法来求解线性系统 $H\beta=T$ 的解 $\hat{\beta}$,如下:

$$\hat{\beta}=H^+T \quad (12)$$

式中, H^+ 就是 Moore_Penrose 广义逆^[11], $\hat{\beta}$ 即为粗糙 RBF 神经网络训练后的隐层与输出层之间的权值.

3.2 粗糙 RBF 神经网络极速学习的算法描述

综上所述,通过前面第 3.1 节中的详细论述,已经确立了粗糙 RBF 神经网络极速学习的一般线性系统数学模型,现将该数学模型的算法实现步骤详细描述如下:

Step1 输入样本的预处理. 根据研究的具体问题,将样本属性处理成为适应粗糙集处理的数据格式,如对连续属性值进行离散化处理等.

Step2 计算样本属性重要度. 依据处理过的样本属性值并结合式(4)计算各属性的知识熵,再利用式(5)进行每个属性的重要度计算.

Step3 利用属性重要度进行样本属性约简. 样本 X 通过属性重要度进行样本属性约简后得到网络的属性约简层输出 $OutPutX$.

Step4 网络参数的初始化. 随机指定属性约简层与隐层之间的连接权值、隐层中心和宽度.

Step5 计算隐层的输出矩阵 H . 利用 Step3 中属性约简层输出的约简样本 $OutPutX$ 与 Step4 中的网络参数再结合式(9)计算整个样本在网络隐层的输出矩阵 H .

Step6 计算隐层与输出层之间的权值. 利用 Step5 中的隐层输出矩阵 H 结合式(12)计算出利用

极速学习机学习的粗糙 RBF 神经网络隐层与输出层之间的权值矩阵 β .

以上 6 个步骤体现出了利用极速学习机学习的粗糙 RBF 神经网络详细学习过程,现就整个网络学习的算法流程图展现如图 2 所示.

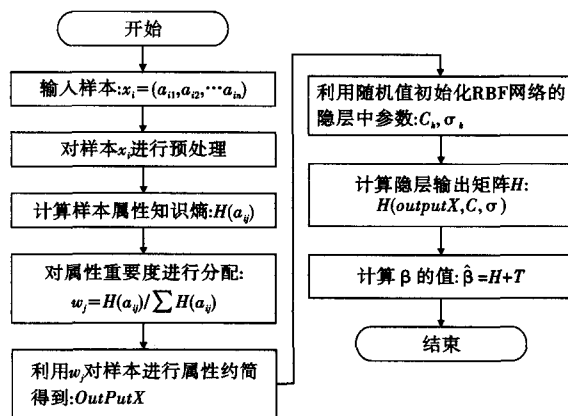


图 2 ELM-RS-RBF 神经网络训练流程图

显然,从算法的流程图可以看出,采用极速学习的粗糙 RBF 神经网络在整个的学习过程中,无需任何的迭代,整个算法将纷繁复杂的迭代运行以及在迭代过程中的网络参数的调节问题一致地转换为了数学公式的计算,极大地提高了整个算法的执行效率.

4 实验分析

为证明本文提出的利用 ELM 方法来训练粗糙 RBF 神经网络的性能,在本实验中分别采用了 UCI 数据集中的三种不同类型数据样本: Wdbc、Wine、Iris 进行了样本分类的训练和测试实验. 在训练测试实验中,ELM 均与 matlab 工具箱自带学习方法(隐层中心聚类法),分别在精度和时间上进行了对比验证. 在实验中,利用 ELM 方法进行粗糙 RBF 神经网络学习时,其网络隐层中心节点为 20, matlab 工具箱自带学习方法中的神经网络隐层中心节点数是根据训练样本个数和预设训练精度从 0 开始递增的,最终的网络隐层中心节点数大于 150. 对三种不同的数据样本均是在相同的实验环境中进行的,实验环境为:硬件系统为 Intel(R) Core(TM)2 CPU T5300,主频 1.73 GHz,内存 1.00 GB;操作系统为 Microsoft Windows XP Professional;编程环境采用 Matlab 7.1.

在进行实验前,分别对这三种数据集做了相同的预处理,即按照决策属性进行分组,将具有相同分

类类别的样本放在一个组中,然后将该数据集的所有组放在同一个集合中;在实验进行中,通过在预处理好的数据集中选择奇数行样本用做训练,偶数行样本用作测试,也即是对每种数据集利用其中一半的数据进行训练,另外一半的数据进行测试;因为数据做了实验前的预处理,所以在进行实验时就能够保证训练测试样本的各种分类的全面覆盖,以达到更好的训练测试效果.这三种不同数据集针对两种不同的学习算法,其各自的实验精度和实验时间如表1、表2、表3所示,表4所示为平均性能.

表1 Wdbc数据集实验结果比较

	训练精度	测试精度	训练时间	测试时间
ELM	0.953 2	0.958 5	0.078 2	0.061 5
Matlab	0.983 1	0.474 3	13.738 1	0.110 3

从表1中非常容易看出,针对于Wdbc数据集,虽然Matlab工具箱的训练精度非常高,但是它的测试精度却是非常的不理想,这是因为Matlab工具箱在进行训练时是根据预定的迭代次数和训练精度两个指标进行不断的迭代,每轮迭代后如果没有达到预定值则会增加一个隐层神经节点,这样一直迭代下去直到满足迭代次数和训练精度中某一个指标才会终止迭代.而ELM却是一种无监督的伪学习,它通过训练样本并借助于式(9)一次性的计算出 $H\beta = T$ 中的 H 矩阵,在 H 矩阵的计算过程中是没有任何监督的,而且也不能称作是一种学习,只是一种纯粹的数学计算,故而简称为伪学习.这也是为什么ELM在训练时间和测试时间上均优越于Matlab工具箱所使用的隐层中心聚类方法.

表2 Wine数据集实验结果比较

	训练精度	测试精度	训练时间	测试时间
ELM	0.943 8	0.932 6	0.031 3	0.015 6
Matlab	0.983 6	0.561 8	0.625 0	0.062 5

在表2的实验结果中可以看出,由于Wine数据集的特征属性比Wdbc数据集的特征属性少了约57%,Wine数据的样本容量比Wdbc的样本容量少了69%,所以两种算法各自在时间效率上的提升是非常显著的,虽然matlab工具箱的训练精度比较高,但是它的测试精度还是非常的不理想.由于样本容量的减小Matlab工具箱训练时间的提升也是可观的.显然,表二的实验数据反映出在样本空间和样本

特征属性减少的情况下,两种算法的训练测试时间会有显著提升.

表3 Iris数据集实验结果比较

	训练精度	测试精度	训练时间	测试时间
ELM	0.985 6	0.959 0	0.030 2	0.015 2
Matlab	0.988 6	0.387 5	0.501 3	0.046 5

在表3中,从Iris数据集得到的实验效果是三种数据集中最好的一种,两种算法其训练精度都比较高,但是matlab工具箱所提供的隐层中心聚类方法,其测试精度仍然偏低,而且还有所下降;ELM的时间效率仍是最高,综合看来针对Iris数据集应用ELM算法训练粗糙RBF神经网络性能最优越.

表4 三种数据集针对两种算法的平均性能结果

	平均训练精度	平均测试精度	平均训练时间	平均测试时间
ELM	0.960 9	0.950 0	0.046 6	0.030 8
Matlab	0.985 1	0.474 5	4.955 4	0.073 1

综合表1~4可知,对这两种不同算法利用三种不同的数据集进行相同实验指标的实验时,非常容易地可以看出ELM的综合性能是最优的,相比于matlab工具箱的平均训练时间效率提升了约106倍,平均测试时间效率提升了约2.4倍,平均测试精度提升了2倍.

5 结束语

本文提出的粗糙RBF神经网络利用ELM学习,具有较高的学习泛化性能,不但学习速度快,而且学习过后的网络泛化精度也是相当高的,针对一般的数据集,其训练时间都是在毫秒级以内,在该实验中,其测试精度比matlab工具箱所采用的隐层中心聚类算法的性能提高了50%以上,这些都足以显示出ELM对粗糙RBF神经网络的极速高精度的学习泛化性能.在接下来的研究中,计划将ELM算法学习应用于粗糙RBF神经网络处理海量数据集的实时效应,比如文本、图像分类等方面,以及如何进一步增强ELM在粗糙RBF神经网络中的学习可控性研究和鲁棒性.

参考文献:

- [1] Xu Xinzhen, Ding Shifei, Shi Zhongzhi, et al. A novel optimizing method for RBF neural network based

- on rough set and AP clustering algorithm[J]. Journal of Zhejiang University-SCIENCE C, 2012, 13(2):131-138.
- [2] Huang G B, Zhu Q Y, Siew C K. Extreme learning machine: theory and applications[J]. Neurocomputing, 2006, 70(1/3):489-501.
- [3] 张文修, 仇国芳. 基于粗糙集的不确定决策[M]. 北京: 清华大学出版社, 2005.
- [4] Pawlak Z, Wong S K M, Ziarko W. Rough Sets: Probabilistic versus deterministic approach[J]. International Journal of Man-Machine Studies, 1988, 29(1):81-95.
- [5] Duntsch I, Gediga G. Uncertainty measures of rough set prediction[J]. Artificial Intelligence, 1998, 106(1):106-109.
- [6] Wierman M J. Measuring uncertainty in rough set theory[J]. International Journal of General Systems, 1999, 28(4):283-283.
- [7] 程林凤, 胡建华. 矩阵论[M]. 徐州: 中国矿业大学出版社, 2009.
- [8] Guang-Bin Huang, Chee Kheong Siew. Extreme learning machine: RBF network case [C] // Automation, Robotics and Vision Conference. Control, 2004 (2):1029-1036.
- [9] Qin Yu Zhu, Qin A K, Suganthan P N, et al. Rapid and brief communication evolutionary extreme learning machine[J]. Pattern Recognition, 2005 (38): 1759-1763.
- [10] Guang-Bin Huang, Xiaojian Ding, Hongming Zhou. Optimization method based extreme learning machine for classification[J]. Neurocomputing, 2010(74):155-163.
- [11] Pierre Courrieu. Fast computation of moore-penrose inverse matrices[J]. Neural Information Processing, 2005, 8(2):25-29.

作者简介:

马刚 男, (1986—), 硕士. 研究方向为机器学习、数据挖掘、神经计算.

丁世飞 男, (1963—), 博士后, 教授, 博士生导师. 研究方向为人工智能、机器学习、数据挖掘、粒度计算等.

史忠植 男, (1941—), 研究员, 博士生导师. 研究方向为智能科学、知识工程、分布智能、机器学习、神经计算、认知科学.

(上接第 8 页)

参考文献:

- [1] Rasmussen C E, Williams C K I. Gaussian processes for machine learning[M]. Boston: the MIT Press, 2006.
- [2] Ranganathan A, Yang Ming-Hsuan, Ho J. Online sparse Gaussian process regression and its applications [J]. IEEE Trans. Image Processing, 2011, 20(2): 391-404.
- [3] 李宏伟, 刘扬, 卢汉清. 结合半监督核的高斯过程分类[J]. 自动化学报, 2009, 35(7): 888-895.
- [4] 王磊, 邹北骥, 彭小宁, 等. 基于高斯过程的表情动作单元跟踪技术[J]. 电子学报, 2007, 35(11): 2087-2091.
- [5] Wang J M, Fleet D J, Hertzmann A. Gaussian process dynamical models for human motion[J]. IEEE Trans. PAMI, 2008, 30(2): 283-298.
- [6] 王雪松, 张依阳, 程玉虎. 基于高斯过程分类器的连续空间强化学习[J]. 电子学报, 2009, 37(6): 1153-1158.
- [7] Jiancheng Sun, Chongxun Zheng, Xiaohe Li, et al. Analysis of the distance between two classes for tuning

SVM hyperparameters[J]. IEEE transactions on neural networks, 2010, 21(2):305-318.

作者简介:



周亚同 男, (1973—), 博士, 副教授. 研究方向为机器学习、模式识别.



贾朋朋 男, 硕士研究生. 研究方向为机器学习、模式识别.

刘龙 男, 副教授. 研究方向为视频图像处理.