

基于 SaCE-ELM 的地铁牵引控制单元快速故障诊断

岳忠奇¹, 吴涛^{1,2}, 顾宏^{*1}

(1. 大连理工大学 控制科学与工程学院, 辽宁 大连 116024;
2. 中车大连电力牵引研发中心有限公司, 辽宁 大连 116052)

摘要: 地铁牵引控制单元(TCU)在地铁运行过程中有重要的作用,及时有效地对其进行故障诊断,是保证地铁正常运行的重要环节.针对传统故障诊断方法的学习速度慢、易陷入局部最优、预测精度较差等缺点,提出一种使用自适应差分进化算法(SaCE)进行优化的极限学习机(SaCE-ELM),即通过自适应差分进化算法对极限学习机的输入权重、隐含层参数和输出权重进行优化.其中,差分进化算法的变异策略通过基于混沌序列的自适应机制产生,其他参数使用正态分布随机生成;网络的输出权重使用 Moore-Penrose 广义逆矩阵计算得出. SaCE-ELM 不需要人工选择变异策略和参数,自适应策略比 SaE-ELM 更加简单.实验结果表明,与 E-ELM、SaE-ELM、LM-NN、SVM 相比, SaCE-ELM 具有更好的预测精度.此外, SaCE-ELM 在所有数据集上训练时间比 SaE-ELM 和 SVM 更少,有效地改善了生成模型的效率.

关键词: 牵引控制单元;故障诊断;极限学习机;差分进化算法

中图分类号: U269.9 **文献标识码:** A **doi:** 10.7511/dllgxb201603008

0 引言

随着我国现代化进程的加快,轨道交通的发展取得了长足的进步.地铁作为各大城市的主要交通工具,其可靠性和安全性受到了多方的重视.在地铁运营过程中,牵引控制单元(traction control unit, TCU)作为控制地铁运行的核心组成部分,通过接收列车的指令信息对整车实施牵引控制.因此,对 TCU 进行快速且有效的故障诊断,是列车保持长期稳定运行的重要环节.

故障诊断的目的是对设备及系统中出现的故障进行检测和识别,从而定位发生故障的部位及其种类.文献[1]将现有的故障诊断方法分为3类:基于分析模型的故障诊断方法、基于定性经验知识的故障诊断方法和基于数据驱动的故障诊断方法.基于分析模型的故障诊断方法需要了解设备的机理结构进而建立精确的定量数学模型,如状态估计法、参数估计法等;基于定性经验

知识的故障诊断方法适用于不易建模的系统,如有向图、专家系统等;基于数据驱动的故障诊断方法是利用设备运行过程中产生的数据信息来进行故障诊断.

对于大多数设备而言,由于其复杂性往往难以建立精确的数学模型,导致基于分析模型的方法不适用.基于定性经验知识的方法需要掌握较深的专业知识,这无疑增加了故障诊断的困难.然而在设备运行的过程中,时刻都会产生大量运行数据,利用这些数据进行故障诊断,是现在许多研究者所关注的.近年来,随着基于数据驱动方法的不断发展,人工智能方法^[2-4](如人工神经网络、支持向量机、模糊逻辑等)在故障诊断中得到了广泛的应用.

极限学习机(extreme learning machine, ELM)是 Huang 等^[5]根据 Moore-Penrose(MP)广义逆矩阵理论提出的一种单隐含层前馈神经网络

络算法. 相比于经典的反向传播 (back propagation, BP) 神经网络^[6]在训练过程中容易陷入局部最优、因反复迭代不能快速且准确获取模型等, ELM 克服了这些缺点, 只需随机指定隐含层参数进而通过最小二乘算法得出输出层参数, 极大地提高了学习速度, 容易满足对预测精度高、结构简单的故障诊断模型快速获取的要求. 由于隐含层节点在训练之前先指定, 最终会导致获得的模型中存在某些对网络性能贡献较少的节点. 而且通常情况下, 相对于其他需要进行节点调整的算法, 极限学习机需要更多的隐含层节点.

本文提出一种改进的自适应差分进化算法 (SaCE), 对极限学习机的输入权重、隐含层偏置和输出权重进行优化. 其中, 进化算法中每一代群体变异策略、交叉因子和缩放因子通过自适应机制进行选择, 进而通过 MP 广义逆矩阵求得输出层参数. 为了验证 SaCE 的有效性, 在 10 个基准函数上和 SaDE^[7-8] 进行比较. 通过和 E-ELM^[9]、SaE-ELM^[10]、LM-NN^[11]、SVM^[12] 在 5 个 UCI 数据集上进行比较验证本文算法 SaCE-ELM 的有效性. 最后, 对 7 种典型的 TCU 故障类型进行定位, 以验证本文算法对 TCU 故障诊断的有效性.

1 极限学习机 (ELM)

对于分类问题, 当给定训练数据集 $\{(x_i, y_i)\}$, 其中 $x_i = (x_{i1} \ x_{i2} \ \cdots \ x_{im}) \subseteq \mathbf{R}^m$ 为示例的特征向量, $y_i = (y_{i1} \ y_{i2} \ \cdots \ y_{im}) \subseteq \mathbf{R}^m$ 为示例标签, $i=1, 2, \dots, N$, 那么对于具有 L 个隐含层节点的单隐含层前馈神经网络 (SLFN) 的输出为

$$o_i = \sum_{j=1}^L \beta_j g(w_j \cdot x_i + b_j); \quad i=1, 2, \dots, N \quad (1)$$

式中: $w_j \in \mathbf{R}^d$ ($j=1, 2, \dots, L$) 是输入层与隐含层第 j 个节点的连接权重, $b_j \in \mathbf{R}$ 是第 j 个隐含层节点偏置参数, $w_j \cdot x_i$ 表示向量内积, $\beta_j \in \mathbf{R}^m$ 是隐含层第 j 个节点与输出层的连接权重. $g(\cdot)$ 为隐含层所采用的输出函数, 如 sigmoid 函数、径向基函数等.

计算输出与实际输出之间的误差即为损失函数:

$$E = \sum_{i=1}^N \left\| \sum_{j=1}^L \beta_j g(w_j \cdot x_i + b_j) - y_i \right\|^2 \quad (2)$$

记 $\theta = (w, b, \beta)$ 代表所有参数, 那么对损失函数最小化, 通常采用基于梯度学习算法将参数 θ 经过反复迭代更新来求解, 更新法则如下:

$$\theta_k = \theta_{k-1} - \eta \frac{\partial E(\theta)}{\partial \theta} \quad (3)$$

其中 η 为学习速率. 虽然反向传播神经网络在众多领域得到了广泛的应用, 但仍然存在如学习速率取值困难、易陷入局部最优、过拟合、收敛速度慢等缺点.

当模型以零误差逼近训练数据集, 即 $\sum_{i=1}^N \|o_i - y_i\|^2 = 0$, 那么存在 w 、 β 和 b , 使得 $\sum_{j=1}^L \beta_j g(w_j \cdot x_i + b_j) = y_i, i=1, 2, \dots, N$, 矩阵形式如下:

$$H\beta = Y \quad (4)$$

式中: $H = H(w, b) = (h_{ij})_{N \times L}$ 为隐含层输出矩阵, $h_{ij} = g(w_j \cdot x_i + b_j)$, β 为输出权重矩阵, $Y = (y_1 \ y_2 \ \cdots \ y_N)$ 为输出矩阵. 不同于传统的梯度下降法需要经过反复迭代调整网络所有参数, 文献[5]指出, ELM 算法只需随机生成参数 w 和 b , 当 w 和 b 固定之后, 求解最小范数最小二乘解即得到输出权重:

$$\beta = H^+ Y \quad (5)$$

其中 H^+ 是 H 的广义逆矩阵.

由此可知, 不同于建立精确的数学模型或专家系统等复杂的系统, 经过训练后的极限学习机网络可以将规则抽象地存储于权重及偏置等网络参数中, 极大地减少了对于设备结构及运行原理的详细了解.

2 差分进化算法及其改进

差分进化算法是由 Storn 等提出的一种进化算法^[13]. 与传统进化算法相同, 它们均有种群初始化、变异、交叉、选择等步骤. 然而, 差分进化算法结构更加简单, 容易实现, 具有更好的全局搜索能力.

基本思想及步骤如下:

(1) 初始化种群

设种群规模为 N_p , 每个个体是 D 维向量. 则第 G 代中的满足上下界的个体可表示为 $x_{i,G} = (x_{i1,G} \ x_{i2,G} \ \cdots \ x_{iD,G}), i=1, 2, \dots, N_p$.

(2) 变异

利用第 G 代不同个体之间的偏差扰动产生第 G 代的候选个体. 如下即为几种使用最多的变异策略, 其中, F 为缩放因子, $K \in [0, 1]$ 为随机生成, $r_1, r_2, r_3, r_4, r_5 \in [1, N_p]$ 为随机产生的不重叠个体序号.

$$\mathbf{v}_{i,G} = \mathbf{x}_{r_1,G} + F \cdot (\mathbf{x}_{r_2,G} - \mathbf{x}_{r_3,G}) \quad (6)$$

$$\mathbf{v}_{i,G} = \mathbf{x}_{r_1,G} + F \cdot (\mathbf{x}_{\text{best},G} - \mathbf{x}_{r_1,G}) + F \cdot (\mathbf{x}_{r_2,G} - \mathbf{x}_{r_3,G}) + F \cdot (\mathbf{x}_{r_4,G} - \mathbf{x}_{r_5,G}) \quad (7)$$

$$\mathbf{v}_{i,G} = \mathbf{x}_{r_1,G} + F \cdot (\mathbf{x}_{r_2,G} - \mathbf{x}_{r_3,G}) + F \cdot (\mathbf{x}_{r_4,G} - \mathbf{x}_{r_5,G}) \quad (8)$$

$$\mathbf{v}_{i,G} = \mathbf{x}_{r_1,G} + K \cdot (\mathbf{x}_{r_1,G} - \mathbf{x}_{r_2,G}) + F \cdot (\mathbf{x}_{r_3,G} - \mathbf{x}_{r_4,G}) \quad (9)$$

研究表明, 变异策略对于不同的优化问题具有不同的效果^[7-8]. 式(6)的全局搜索能力强, 但收敛速度较慢; 式(7)局部搜索能力强并且能够快速收敛, 但容易陷入局部最优从而导致早熟; 式(8)比式(6)搜索能力更强, 但计算量随之增大; 式(9)可以有效地解决多目标优化问题.

(3) 交叉

将变异个体 $\mathbf{v}_{i,G+1}$ 与当前个体 $\mathbf{x}_{i,G}$ 进行交叉, 得到当前个体的候选个体 $\mathbf{u}_{i,G+1}$.

$$\mathbf{u}_{ij,G} = \begin{cases} \mathbf{v}_{ij,G}; & \text{rand} < R_c \text{ 或 } j = R(i) \\ \mathbf{x}_{ij,G}; & \text{其他} \end{cases} \quad (10)$$

其中 $i = 1, 2, \dots, N_p, j = 1, 2, \dots, D, \text{rand} \in [0, 1]$ 是一个均匀分布随机数, $R_c \in [0, 1]$ 为交叉概率, $R(i)$ 是 $[1, D]$ 内的随机整数, 采用这种方法可确保下一代个体中至少有一条染色体来自变异个体.

(4) 选择

通过对第 G 代个体 $\mathbf{x}_{i,G}$ 以及候选个体 $\mathbf{u}_{i,G+1}$ 适应度的评价, 根据下式决定个体的进化方向:

$$\mathbf{x}_{i,G+1} = \begin{cases} \mathbf{u}_{i,G+1}; & f(\mathbf{u}_{i,G+1}) \leq f(\mathbf{x}_{i,G}) \\ \mathbf{x}_{i,G}; & \text{其他} \end{cases} \quad (11)$$

虽然差分进化算法已经在众多领域取得了广泛的应用, 但它仍有不可避免的缺点. 比如, 对于特定的问题, 选择最佳的变异策略尤其困难. 文献[14]将每种策略的概率参数预先指定, 从而进行自动的策略选取; 文献[7-8]所提出的 SaDE 算法根据前 G 代变异策略的使用情况获得选择每种策略的概率; 文献[10]成功地将 SaDE 应用于

ELM 产生 SaE-ELM 并且取得了良好的实验效果, 但程序编码过程较为烦琐; 文献[15]提出的多策略差分进化算法, 采用父个体作为索引的方式从策略库中选取策略; 文献[16]提出了一种简单的多策略自适应选择机制, 通过设定自适应策略参数实现种群中每个个体的策略选取.

文献[16]中策略计算方式如下:

$$S_i = \lfloor \delta_i \times K \rfloor + 1 \quad (12)$$

其中 $\delta_i \in [0, 1)$ 为策略参数, 文献[16]提出了 3 种策略参数的选择方法, K 为策略数量. 如 $\delta_i \in (0, 0.25)$ 且 $K = 4$, 则 $S_i = 1$, 表明当前个体 \mathbf{x}_i 应选择第一种策略.

不同于文献[12]提出的策略参数自适应机制, 本文提出一种基于混沌序列的参数自适应机制. 混沌序列 (chaotic sequence) 具有随机性、遍历性、普适性等特点. 最近, 一些学者成功地将混沌序列与进化算法结合应用到了优化问题中, 并取得了较好的结果. 文献[17]将混沌序列作为进化算法控制参数选择的策略; 文献[4, 18]将混沌序列作为进化算法种群初始化的有利工具.

作为混沌序列的典型模型, 逻辑映射公式如下:

$$\delta_{k+1} = \mu \delta_k (1 - \delta_k); \quad k = 1, 2, \dots, N \quad (13)$$

其中 $\delta \in (0, 1), \delta \neq 0.25, 0.50, 0.75, k$ 为迭代次数, μ 为混沌参数, 设 $\mu = 4$.

本文将上述逻辑映射作为式(12)中策略参数 δ_i 的生成机制. 根据上文所述可知, 单一的变异策略在很多情况下无法满足要求, 故而采用多策略相互配合能够获得更满意的效果. 选取式(6)~(9) 4 种策略作为本文策略池. 缩放因子通过正态分布 $N(0.7, 0.3)$ 随机产生. 对于交叉概率, 首先通过多次迭代保存较优值, 待达到一定迭代次数后, 将之前保存的较优值取均值 R_{cmean} , 之后的新种群中根据正态分布 $N(R_{\text{cmean}}, 0.1)$ 随机生成每个个体的交叉概率. 综上所述, 本文提出了基于混沌序列的自适应差分进化算法 SaCE.

3 自适应差分进化极限学习机 (SaCE-ELM)

为了克服 E-ELM 人工选择变异策略的产生耗时、选择不当等问题, 采用基于混沌序列的自适

应策略机制来实现差分进化算法的多策略自适应选取,并将其应用到优化极限学习机的网络参数中.

给定数据集 $\{(x_i, y_i)\}, i=1, 2, \dots, N$, 将其分为训练数据集、验证数据集、测试数据集, 隐含层节点个数为 L , 激活函数 $g(\cdot)$ 采用 sigmoid 函数, 种群数量为 N_p , 指定缩放因子 F 和交叉概率 R_c 的初始值, 随机生成符合定义域策略参数 δ , 归纳 SaCE-ELM 算法步骤如下:

(1) 初始化种群

将输入权重和隐含层偏置作为个体进行种群初始化, 个体如下:

$$\mathbf{x}_{i,G} = (w_{1,G}^T \quad \dots \quad w_{L,G}^T \quad b_{1,G} \quad \dots \quad b_{L,G})$$

其中 w_j 和 $b_j (j=1, 2, \dots, L)$ 在范围 $[-1, 1]$ 内随机初始化, G 代表种群代数, $i=1, 2, \dots, N_p$ 代表种群个体序号.

(2) 计算输出权重及个体适应度

通过广义逆矩阵代替传统的迭代方式计算输出权重 $\beta_{i,G} = \mathbf{H}_{i,G}^+ \mathbf{Y}$; 将验证数据集分类错误率作为个体适应度评价准则 $f(\mathbf{x}_{i,G}) = 1 - A_{cc}$, A_{cc} 为分类准确率. 在第一代种群中, 将适应度最优的个体作为最佳个体 $\mathbf{x}_{1,best}$ 进行存储.

(3) 变异和交叉

通过自适应策略分别对每个个体进行变异策略的选择, 缩放因子和交叉概率根据上文所述方法进行选取, 进而获得候选个体 $\mathbf{u}_{i,G+1}$.

(4) 选择

文献[19]指出, 对于前馈型神经网络, 权重越小, 其预测精度越好. 文献[9]使用标准差分进化算法优化极限学习机的所有参数, 称为 E-ELM 算法. 该算法的个体进化准则除标准的适应度评价之外, 还包括输出权重 β 的 2-范数 $\|\beta\|$. 研究表明, 该方法更有利于优质个体的筛选.

$$\mathbf{x}_{i,G+1} = \begin{cases} \mathbf{u}_{i,G+1}; & f(\mathbf{x}_{i,G}) - f(\mathbf{u}_{i,G}) > \epsilon \cdot f(\mathbf{x}_{i,G}) \text{ 或} \\ & |f(\mathbf{x}_{i,G}) - f(\mathbf{u}_{i,G})| > \epsilon \cdot f(\mathbf{x}_{i,G}) \\ & \text{且 } \|\beta_{\mathbf{u}_{i,G+1}}\| < \|\beta_{\mathbf{x}_{i,G}}\| \\ \mathbf{x}_{i,G}; & \text{其他} \end{cases} \quad (14)$$

其中 $f(\mathbf{x}_{i,G})$ 和 $f(\mathbf{u}_{i,G})$ 分别为父个体和候选个体适应度. 不断重复步骤(3)和(4)直到达到适应度目标值或者最大迭代次数. 算法流程图如图 1 所示.

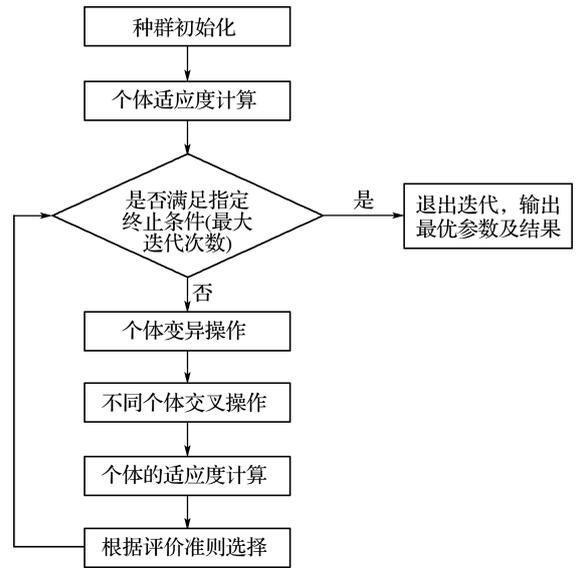


图 1 SaCE-ELM 算法流程图

Fig. 1 Flow diagram of SaCE-ELM algorithm

4 算法有效性评估

4.1 基准函数实验

基于表 1 中列出的 10 个基准函数, 对自适应差分进化算法 SaCE 与 SaDE 进行比较, 其中每个函数均在维数 $D=30$ 进行测试, 对每个函数分别进行 5 次实验后取平均值, 并列出最优值、最差值、均值、标准差, 以均值为评判标准进行比较, 实验结果见表 2.

实验结果表明, 与 SaDE 相比较, SaCE 在多数基准函数上的优化效果较好. 其中 f_{04} 函数上表现比 SaDE 差, f_{01} 、 f_{07} 、 f_{08} 、 f_{09} 函数上表现与 SaDE 基本相同, f_{02} 、 f_{03} 、 f_{05} 、 f_{06} 、 f_{10} 函数上表现优于 SaDE. SaDE 所用优化时间 $t=3\ 867.73\ \text{s}$, SaCE 优化时间 $t=3\ 125.96\ \text{s}$, 可见, SaCE 所用优化时间明显少于 SaDE.

4.2 UCI 分类数据集实验

为了验证分类器的有效性, 本文从 UCI 数据库选取 5 个典型的分类数据集 Disease、Diabetes、Iris、Wine、Segment 进行实验验证. 首先将所有数据集的属性标准化到 $[-1, 1]$. 然后将数据集分成训练数据集和测试数据集, 之后选取训练数据集的 30% 作为验证数据集. 数据集的详细信息见表 3.

表 1 基准函数详述

Tab. 1 Specifications of benchmark functions

函数名	测试函数	定义域
Sphere	$f_{01} = \sum_{i=1}^D x_i^2$	$[-100, 100]^D$
Schwefel 2. 22	$f_{02} = \sum_{i=1}^D x_i + \prod_{i=1}^D x_i $	$[-10, 10]^D$
Schwefel 1. 2	$f_{03} = \sum_{i=1}^D \left(\sum_{j=1}^i x_j \right)^2$	$[-100, 100]^D$
Schwefel 2. 21	$f_{04} = \max \{ x_i , 1 \leq i \leq D \}$	$[-100, 100]^D$
Rosenbrock	$f_{05} = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2) + (x_i - 1)^2]$	$[-30, 30]^D$
Step	$f_{06} = \sum_{i=1}^{D-1} \lfloor x_i + 0.5 \rfloor^2$	$[-100, 100]^D$
Quartic	$f_{07} = \sum_{i=1}^D x_i^4 + \text{random}[0, 1)$	$[-1.28, 1.28]^D$
Schwefel 2. 26	$f_{08} = \sum_{i=1}^D (-x_i \sin(\sqrt{ x_i }))$	$[-500, 500]^D$
Rastrigin	$f_{09} = \sum_{i=1}^D (x_i^2 - 10 \cos(2\pi x_i) + 10)$	$[-32, 32]^D$
Ackley	$f_{10} = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^D x_i^2}\right) - \exp\left(\frac{1}{D} \sum_{i=1}^D \cos(2\pi x_i)\right) + 20 + \exp(1)$	$[-600, 600]^D$

表 2 基准函数实验结果

Tab. 2 Experimental results for benchmark functions

基准函数	SaCE				SaDE			
	最优值	最差值	均值	标准差	最优值	最差值	均值	标准差
f_{01}	0	0	0	0	0	0	0	0
f_{02}	1.12×10^{-6}	3.74×10^{-5}	1.29×10^{-5}	1.55×10^{-5}	3.00×10^{-8}	6.87×10^{-4}	1.42×10^{-4}	3.05×10^{-4}
f_{03}	9.69×10^4	5.62×10^5	3.58×10^5	1.73×10^5	3.56×10^5	5.94×10^5	4.64×10^5	8.72×10^4
f_{04}	1.10×10^1	9.44×10^2	2.63×10^2	3.95×10^2	1.15×10^1	2.18×10^2	8.07×10^1	8.11×10^1
f_{05}	2.81×10^3	3.14×10^3	2.97×10^3	1.45×10^2	3.41×10^3	4.16×10^3	3.76×10^3	3.08×10^2
f_{06}	1.03×10^0	9.95×10^1	4.38×10^1	4.10×10^1	1.95×10^1	8.36×10^1	4.71×10^1	3.32×10^1
f_{07}	4.69×10^3	4.69×10^3	4.69×10^3	0	4.69×10^3	4.70×10^3	4.69×10^3	6.43×10^{-3}
f_{08}	2.08×10^1	2.10×10^1	2.09×10^1	8.94×10^{-2}	2.08×10^1	2.10×10^1	2.09×10^1	5.79×10^{-2}
f_{09}	0	0	0	0	0	0	0	0
f_{10}	2.98×10^1	4.37×10^1	3.74×10^1	6.31×10^0	3.38×10^1	5.97×10^1	4.66×10^1	1.15×10^1

表 3 UCI 分类数据集详述

Tab. 3 Specifications of UCI classification datasets

数据集	样本数		属性	类别
	训练数据集	测试数据集		
Disease	100	170	13	2
Diabetes	500	268	8	2
Iris	100	50	4	3
Wine	100	78	13	3
Segment	1 910	1 000	18	7

对以上数据集,所有算法均进行多次实验后取平均值,实验结果及参数选取见表 4. 对于算法

SaCE-ELM、SaE-ELM、E-ELM,每次实验种群 N_p 设置为 20,初始缩放因子 F_0 设置为 1,初始交叉概率 R_{c0} 设置为 0.8,每种算法的隐含层节点首先在较小范围 $[5, 20]$ 内随机生成,之后根据实验结果依次递增,保留最佳节点个数.其中,E-ELM 人工选择 4 种变异策略依次实验,保留最优实验结果.LM-NN 通过最小二乘法实现反向传播神经网络,隐含层节点选取方法同上,使用 Matlab 的神经网络工具箱进行实验.SVM 采用径向基函数(RBF)作为核函数,惩罚因子 C 和核参数 γ 在

范围 $\{2^{12}, 2^{11}, \dots, 2^{-2}\}$ 和 $\{2^4, 2^3, \dots, 2^{-10}\}$ 内采用网格搜索法进行最佳参数选取, 从而提高分类器的预测精度. 如表 4 所示, 实验结果列出预测精度、训练时间、节点/支持向量个数.

表 4 UCI 数据集实验结果比较

Tab. 4 Experimental results comparison for UCI datasets

数据集	算法	预测精度		训练时间/s	节点/支持向量
		均值	标准差		
Disease	SaCE-ELM	82.88	1.61	0.493 3	10
	SaE-ELM	82.53	3.86	0.708 6	18
	E-ELM	81.62	4.29	0.224 7	16
	LM-NN	71.75	6.67	0.306 6	20
	SVM	76.10	3.46	6.709 4	81.6
Diabetes	SaCE-ELM	81.92	2.37	3.232 0	10
	SaE-ELM	79.55	2.68	5.642 2	20
	E-ELM	78.09	3.53	1.475 0	30
	LM-NN	74.73	3.20	2.897 8	20
	SVM	77.31	2.56	364.6	62.28
Iris	SaCE-ELM	98.27	2.39	0.370 0	15
	SaE-ELM	97.20	4.12	0.228 9	18
	E-ELM	96.48	3.67	0.152 2	16
	LM-NN	95.40	3.19	0.364 1	10
	SVM	94.36	2.26	4.548 8	23.3
Wine	SaCE-ELM	98.23	2.78	0.228 8	15
	SaE-ELM	97.95	2.54	0.305 0	22
	E-ELM	97.18	2.44	0.210 0	20
	LM-NN	92.97	4.36	0.498 8	20
	SVM	97.48	1.57	5.894 1	47.3
Segment	SaCE-ELM	95.92	1.57	44.359 4	90
	SaE-ELM	95.58	1.21	256.4	90
	E-ELM	95.27	1.56	154.1	70
	LM-NN	86.27	1.95	4 745.7	100
	SVM	93.58	1.65	802.3	271.6

实验结果表明, 相比于 SaE-ELM、E-ELM、LM-NN 和 SVM, 本文算法 SaCE-ELM 在所有实验数据集上均取得最高的预测精度. 与本文算法相比, SaE-ELM 编码过程比本文算法烦琐, 而且训练时间较长. 在 4 个较小的数据集 Disease、Diabetes、Iris 和 Wine 上, E-ELM 虽然训练时间最短, 但是其需要通过人工选择 4 种变异策略保留最优值, 因而实际耗时明显多于 SaCE-ELM. 采用最小二乘法实现的 BP 神经网络 LM-NN, 虽然在数据量较小的数据集上训练时间有了明显改善, 但在较大的数据集 Segment 上训练时间远远大于 SaCE-ELM 等算法, 并且预测精度较差. SVM 虽然稳定性较好, 但从实验结果可以得出, 随着数据量的增大, 其训练时间增长较快.

5 地铁牵引控制单元故障诊断

5.1 故障诊断流程

列车控制及监控系统 (TCMS) 依据 IEC61375 标准规定的列车通信网络组建, 该系统具有牵引、制动等控制功能. 其中, 牵引控制单元通过接收 TCMS 系统发送的指令信息, 实现对整车牵引系统的控制. 列车故障诊断功能由位于 TC 车的诊断中央控制单元完成. 该诊断系统具有故障信息识别以及故障信息输出功能.

对于牵引控制单元而言, 由于设备复杂而难于建立精确的数学诊断模型. 同时, 只有少数专家能够通过较深的专业知识进行故障诊断. 然而, 在设备工作的过程中, 会不断产生大量的输入输出数据, 同时数据存储设备会将所有数据进行存储. 因此, 基于数据分析进行故障诊断成为了一种较好的选择.

牵引控制单元故障诊断流程如图 2 所示, 其中 MVB 信息流即为通过 MVB 总线传输的各个数字量与模拟量. 首先对信息流进行筛选, 将牵引控制单元接收的输入信息流进行特征提取、信息融合后进行合理的数据处理, 通过智能算法进行故障诊断后, 故障数据最终显示到人机交互界面, 同时, 通过指定格式进行数据存储, 以便分析.

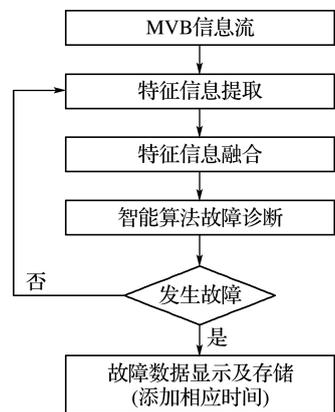


图 2 牵引控制单元故障诊断流程图

Fig. 2 Flow diagram of fault diagnosis of TCU

5.2 数据提取

文献[12]将与牵引控制单元有关的指令信息作为样本的特征, 包含 14 个数字量和模拟量, 本文选取最终输入牵引控制单元的 11 个变量作为样本特征, 如图 3 所示. 在牵引系统运行过程中,

如果发生故障,会产生牵引系统隔离的现象,最终导致牵引系统失去作用.将重要故障分为7类,即HSCB跳闸、电机电流超过2 200 A、逆变器故障、牵引电机警告级高温、滤波电压超过2 150 V、380 V供电故障和荷载信号故障.

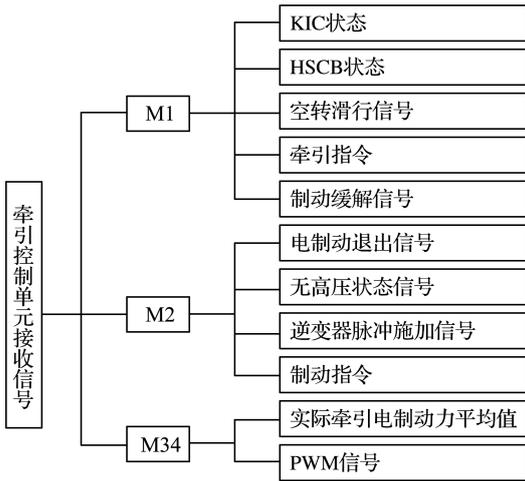


图3 牵引控制单元输入变量

Fig.3 Input variables of TCU

机车在运行过程中会不断产生历史数据和新生数据,本文使用中车大连电力牵引研发中心有限公司研发的终端维护软件PTU对已保存的故障样本数据进行特征及类别提取.其中,特征变量分别位于M存储区的M1、M2、M34区域.详细数据提取流程如图4所示,虚框A和B分别为故障类别以及故障特征的生成过程,最后合并成故障样本数据.其中,故障数据保存于failure.csv文件中,故障发生时间及其指令信息保存于eventdata.csv文件中.

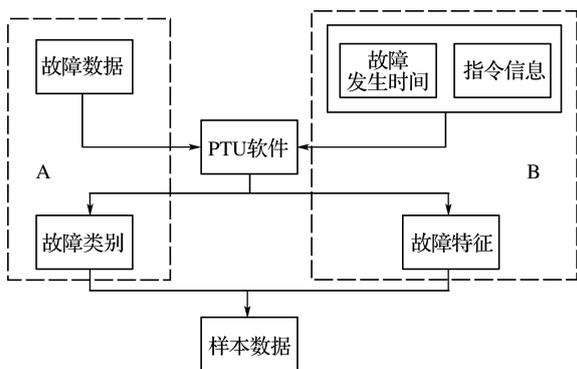


图4 牵引控制单元故障数据提取流程图

Fig.4 Flow diagram of TCU fault data extraction

本文根据上述数据提取方式,使用PTU软件对故障数据文件和指令信息文件提取相应数据集,取得551组数据作为实验数据集 $\{(x_i, y_i)\}$,其中 $x_i=(x_{i1} \ x_{i2} \ \dots \ x_{i11}) \subseteq \mathbf{R}^{11}$ 为进行信息融合后的11维故障特征, $y_i=(y_{i1} \ y_{i2} \ \dots \ y_{i7}) \subseteq \mathbf{R}^7$ 为7类故障类别.各类别样本均匀分布.将所有样本属性归一化到 $[-1, 1]$,随机从原始数据集中取得371个样本作为训练集,取得180个样本用作测试集,再取训练集的30%作为验证数据集用作本文算法中适应度函数的评价样本集.故障诊断模型经过反复训练后,得出 w 、 b 、 β 分别作为极限学习神经网络的输入权重、隐含层偏置、输出权重,对于故障特征和故障类别的对应关系,通过极限学习神经网络进行非线性映射.

5.3 实验结果及分析

所有算法参数设置方法与UCI数据集实验基本相同.实验结果见表5,表中列出预测精度、训练时间以及节点/支持向量个数.

表5 TCU故障数据集实验结果比较

Tab.5 Experimental results comparison for TCU fault dataset

数据集	算法	预测精度		训练时间/s	节点/支持向量
		均值	标准差		
TCU Fault	SaCE-ELM	93.33	1.48	3.012 5	15
	SaE-ELM	92.12	1.55	5.891 2	20
	E-ELM	88.79	2.21	2.231 1	30
	LM-NN	86.56	2.70	3.175 1	20
	SVM	92.21	1.42	50.78	42

实验结果表明,相比其他算法,SaCE-ELM算法在TCU故障数据集上具有更好的预测精度以及较短的训练时间.LM-NN的训练时间与SaCE-ELM较为接近,但是预测精度较差.通过人工选择E-ELM的4种变异策略并保留最佳实验结果后,E-ELM的训练时间最短,但需要额外的人工选择时间.与SaE-ELM相比,SaCE-ELM采用较为简单的编码策略实现了自适应策略,训练时间明显降低,快速地生成了所需模型.虽然SVM的预测精度与本文算法较为接近,但训练时间远远大于本文算法.总而言之,将本文算法应用于TCU故障诊断,取得了良好的实验结果,验证了本文算法的有效性.

6 结 语

本文提出一种基于混沌序列的自适应差分进化算法进行优化的极限学习机 SaCE-ELM, 作为地铁牵引控制单元故障分类模型。不同于标准差分进化算法的人工选择变异策略以及参数, SaCE-ELM 采用基于混沌序列的自适应机制有效地对参数以及变异策略进行了自适应选择, 而且编码过程较为简单。实验结果表明, 本文算法有效地减少了训练时间并且改善了模型的预测精度。最后, 将本文算法应用到地铁牵引控制单元故障数据上, 结果表明本文算法能有效地对地铁牵引控制单元进行故障诊断。

参考文献:

- [1] 李 晗, 萧德云. 基于数据驱动的故障诊断方法综述[J]. 控制与决策, 2011, **26**(1):1-9.
LI Han, XIAO De-yun. Survey on data driven fault diagnosis methods [J]. **Control and Decision**, 2011, **26**(1):1-9. (in Chinese)
- [2] Daley S, Newton D A, Bennett S M, *et al.* Methods for fault diagnosis in rail vehicle traction and braking systems [J]. **IEEE Colloquium (Digest)**, 1995(79):1-13.
- [3] Chen J, Roberts C, Weston P. Fault detection and diagnosis for railway track circuits using neuro-fuzzy systems [J]. **Control Engineering Practice**, 2008, **16**(5):585-596.
- [4] 徐晓璐, 吴 涛, 顾 宏. 基于 IPSO-SVM 的地铁车辆牵引控制单元故障诊断[J]. 大连理工大学学报, 2015, **55**(1):67-72.
XU Xiao-lu, WU Tao, GU Hong. Fault diagnosis of metro vehicle traction control unit based on IPSO-SVM [J]. **Journal of Dalian University of Technology**, 2015, **55**(1):67-72. (in Chinese)
- [5] Huang G B, Zhu Q Y, Siew C K. Extreme learning machine: A new learning scheme of feedforward neural networks [J]. **IEEE International Conference on Neural Networks - Conference Proceedings**, 2004, **2**:985-990.
- [6] WANG Lin, ZENG Yi, CHEN Tao. Back propagation neural network with adaptive differential evolution algorithm for time series forecasting [J]. **Expert Systems with Applications**, 2015, **42**(2):855-863.
- [7] Qin A K, Suganthan P N. Self-adaptive differential evolution algorithm for numerical optimization [C] // **2005 IEEE Congress on Evolutionary Computation, IEEE CEC 2005. Proceedings**. Edinburgh: IEEE Computer Society, 2005: 1785-1791.
- [8] Qin A K, Huang V L, Suganthan P N. Differential evolution algorithm with strategy adaptation for global numerical optimization [J]. **IEEE Transactions on Evolutionary Computation**, 2009, **13**(2):398-417.
- [9] Zhu Q Y, Qin A K, Suganthan P N, *et al.* Evolutionary extreme learning machine [J]. **Pattern Recognition**, 2005, **38**(10):1759-1763.
- [10] Cao J W, Lin Z P, Huang G B. Self-adaptive evolutionary extreme learning machine [J]. **Neural Processing Letters**, 2012, **36**(3):285-305.
- [11] Hagan M T, Menhaj M B. Training feedforward networks with the Marquardt algorithm [J]. **IEEE Transactions on Neural Networks**, 1994, **5**(6):989-993.
- [12] Hsu C W, Lin C J. A comparison of methods for multiclass support vector machines [J]. **IEEE Transactions on Neural Networks**, 2002, **13**(2):415-425.
- [13] Storn R, Price K. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces [J]. **Journal of Global Optimization**, 1997, **11**(4):341-359.
- [14] Zamuda A, Brest J, Bošković B, *et al.* Large scale global optimization using differential evolution with self-adaptation and cooperative co-evolution [C] // **2008 IEEE Congress on Evolutionary Computation, CEC 2008**. Piscataway: IEEE Computer Society, 2008:3718-3725.
- [15] 俞国燕, 李 鹏, 何 真, 等. 带自适应动态变异和二次变异的差分进化算法[J]. 计算机集成制造系统, 2010, **16**(5):987-993.
YU Guo-yan, LI Peng, HE Zhen, *et al.* Differential evolution with adaptive dynamic mutation & second mutation [J]. **Computer Integrated Manufacturing Systems**, 2010, **16**(5): 987-993. (in Chinese)
- [16] GONG Wen-yin, CAI Zhi-hua, Ling C X, *et al.*

- Enhanced differential evolution with adaptive strategies for numerical optimization [J]. **IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics**, 2011, **41**(2):397-413.
- [17] CHEN Guang-yu, DING Xiao-qun. An improved differential evolution method based on the dynamic search strategy to solve dynamic economic dispatch problem with valve-point effects [J]. **Abstract and Applied Analysis**, 2014, **2014**:175417.
- [18] Caponetto R, Fortuna L, Fazzino S, *et al.* Chaotic sequences to improve the performance of evolutionary algorithms [J]. **IEEE Transactions on Evolutionary Computation**, 2003, **7**(3):289-304.
- [19] Bartlett P L. Sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network [J]. **IEEE Transactions on Information Theory**, 1998, **44**(2):525-536.

Fast fault diagnosis of metro traction control unit based on SaCE-ELM

YUE Zhong-qi¹, WU Tao^{1,2}, GU Hong^{*1}

(1. School of Control Science and Engineering, Dalian University of Technology, Dalian 116024, China;
2. CRRC Dalian R&D Co., Ltd., Dalian 116052, China)

Abstract: Metro traction control unit (TCU) plays a key role in the operation of subway. It is important for the normal operation of subway to diagnose the TCU fault timely and effectively. However, the traditional fault diagnosis methods usually have some disadvantages, such as slow learning speed, falling into local optimum easily and poor prediction accuracy. To solve these problems, extreme learning machine based on adaptive differential evolution algorithm (SaCE-ELM) is proposed. The input weights, the implicit layer parameters and the output weights of the extreme learning machine are optimized by adaptive differential evolution algorithm. The variation strategy of differential evolution algorithm is generated by the adaptive mechanism based on chaotic sequence, and other parameters are randomly generated using normal distribution. The output weights of the network are calculated using Moore-Penrose generalized inverse matrix. SaCE-ELM doesn't need artificial selection of variation strategy and parameters, and its adaptive strategy is simpler than that of SaE-ELM. Experimental results show that SaCE-ELM has better prediction accuracy compared with E-ELM, SaE-ELM, LM-NN and SVM. Moreover, the training time of SaCE-ELM is shorter than that of SaE-ELM and SVM in all experimental datasets, which demonstrates that the efficiency of model generation has been improved.

Key words: traction control unit; fault diagnosis; extreme learning machine; differential evolution algorithm