doi:10.11835/j.issn.1000-582X.2021.01.004

一种基于凸包稀疏化与遗传算法优化的 SVM 算法

钱红兵,李艳丽
(中国人民大学 信息技术中心,北京 100089)

摘要:SVM(support vector machine)算法求解支持向量的过程涉及到 N 阶矩阵的计算,N 为样本的个数,当样本数量很大时,高阶矩阵的计算将需要消耗大量运算时间;同时,SVM 模型性能依赖于惩罚参数与核参数的优化,传统的循环验证参数优化法,时间复杂度高。为了解决上面两方面的问题,笔者采用凸包算法对训练样本进行稀疏化,同时通过遗传算法优化选择惩罚参数与核参数,提出了一种高性能的 SVM 模型训练算法。

关键词:遗传算法;凸包算法;支持向量机;惩罚参数;核参数
中图分类号:TP311 **文献标志码**:A **文章编号**:1000-582X(2020)01-029-08

A SVM algorithm based on convex hull sparsity and genetic algorithm optimization

QIAN Hongbing, *LI Yanli*
(Information Technology Center, Renmin University of China, Beijing 100089, P. R. China)

Abstract: SVM algorithm for support vector involves the calculation of N-order matrix. N is the number of samples. When the number of samples is large, the calculation of high-order matrix will consume a lot of computing time. At the same time, the performance of SVM model depends on the optimization of penalty parameters and kernel parameters. The traditional cycle verification method of parameter optimization has high time complexity. In order to solve these two problems, this paper proposes a high-performance SVM model training algorithm by convex hull algorithm to sparse the training samples and by genetic algorithm to optimize the selection of penalty parameters and kernel parameters.

Keywords: genetic algorithm; convex hull; SVM; penalty parameter; kernel parameter

使用大规模样本数据训练 SVM 模型时存在计算效率问题,对样本进行稀疏化,能有效减少 SVM 模型的训练时间,但样本稀疏化后,一些支持向量也会被删除,数据会出现失真,进而影响模型的准确度。如何在尽量保证模型准确度的情况下进行样本数据的稀疏化,是一个待研究问题。现在有使用随机采样的方法,也有使用全局代表点替代的方法,但都会造成支持向量的缺失。通过研究 SVM 算法的特点,利用支持向量点和凸包顶点的相似性,引入计算机图形学的凸包算法,通过求解训练数据集的凸包点,获取最多的 SVM 分类支持向量,排除非支持向量,在尽可能保证训练样本数据特征的情况下,实现样本数据的稀疏化,进而降低

收稿日期:2020-04-14
基金项目:中国高等教育学会重点资助项目(2019ZXZD011)。
Supported by the Key Commissioned Project of China Association of Higher Education in (2019ZXZD011).
作者简介:钱红兵(1983-),男,硕士,主要从事软件工程方向研究,(E-mail)qianhb@ruc.edu.cn。
通讯作者:李艳丽,女,博士,副研究员,主要从事教育信息化方向研究,(E-mail)liyl@ruc.edu.cn。

SVM模型的训练时间。

同时使用径向基核函数(RBF, radial basis function)的SVM模型分类精度受到核参数 g 和惩罚因子 c 的影响较大,如果参数设置不当,SVM分类的精度将会下降[1]。如何获取最优参数是一个优化求解问题,现在常采用交叉验证法,这种方法需要同时验证 g 和 c 的在给定取值范围中的所有组合,算法的时间复杂度高,效率低,为了快速获取SVM分类的最优参数,引入遗传算法进行 g 和 c 的优化求解,进一步降低了SVM模型的训练时间。

通过上面2方面的创新和改进,提出了改进优化的SVM训练算法,在保证SVM模型分类精度的情况下,提高SVM模型训练效率。

1 SVM算法

SVM支持向量机是一种二分类模型,在机器学习分类算法中,SVM是应用很广的一种算法[2],其算法的本质,是通过将样本数据映射到一个高维空间中,然后在空间中寻找一个超平面,这个超平面能够准确地分开2类数据样本并使得其间隔最大化。这个超平面称为分离超平面。

如图1所示,分离超平面可以使用公式表示为

$$H:\boldsymbol{W}^{\mathrm{T}}\cdot\boldsymbol{X}+b=0,$$

其中,$\boldsymbol{W}^{\mathrm{T}}$ 是权重向量,b 是偏移。这样位于分离超平面两侧的点可以表示为

$$\boldsymbol{W}^{\mathrm{T}}\cdot\boldsymbol{X}+b>0 \text{ 或者 } \boldsymbol{W}^{\mathrm{T}}\cdot\boldsymbol{X}+b<0,$$

可以调整权重使得2个边缘超平面定义为

$$H1:\boldsymbol{W}^{\mathrm{T}}\cdot\boldsymbol{X}+b=-1,$$

$$H2:\boldsymbol{W}^{\mathrm{T}}\cdot\boldsymbol{X}+b=1。$$

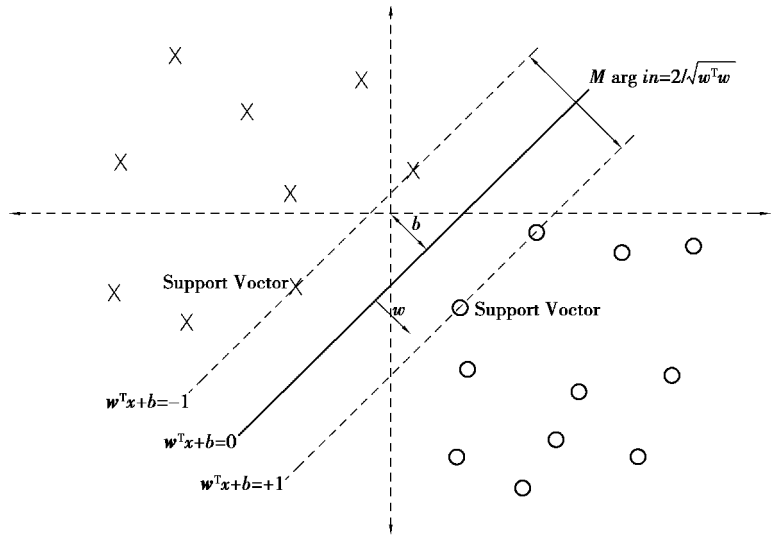

图1 SVM算法示意图

Fig. 1 Schematic diagram of SVM algorithm

这样所有位于 $H1$ 平面上方的点都被归类为-1类型,所有位于 $H2$ 平面下方的点都被归类为1类型,所有位于 $H1$ 或者 $H2$ 平面上面的样本点被称为支持向量。从分离超平面到 $H1$ 的距离是$\frac{1}{\sqrt{\boldsymbol{W}^{\mathrm{T}}\boldsymbol{W}}}$,$H1$ 到 $H2$ 的距离也叫最大边缘距离为$\frac{2}{\sqrt{\boldsymbol{W}^{\mathrm{T}}\boldsymbol{W}}}$。最大化$\frac{2}{\sqrt{\boldsymbol{W}^{\mathrm{T}}\boldsymbol{W}}}$,等价于最小化$\frac{\|\boldsymbol{W}\|^2}{2}$,并满足约束条件 $y_i(\boldsymbol{W}^{\mathrm{T}}x^i+b)\geqslant 1$,可以将问题转换为优化模型

$$\begin{cases}\min\Phi(\omega)=\dfrac{1}{2}\|\omega\|^2+c\left(\sum\limits_{i=1}^{N}\xi_i\right),\\ y_i(\omega\Phi(x_i)+b)>=1-\xi_i,\xi_i>=0。\end{cases}$$

使用拉格朗日公式和 KTT 条件优化公式，最后问题转换为求解 L 最大值问题

$$\begin{cases}L_{\max}=\sum\limits_{i=1}^{N}a_i-\dfrac{1}{2}\sum\limits_{i=1}^{N}\sum\limits_{j=1}^{N}y_iy_ja_ia_jexp(-g\|x_i-x_j\|^2),\\ \sum\limits_{i=1}^{N}a_iy_i=0,a_i>=0。\end{cases}$$

核参数 g 的取值影响模型的分类精度，惩罚因子 c 影响模型的推广能力，如果 c 取值过大 SVM 分类模型将会过拟合[1-6]。

2　Convex Hull

凸包(convex hull)是一个源自于计算机图形学的概念，其数学定义为：在一个 N 维向量空间 Q 中，对于一个给定的集合 P，所有包含 P 的凸集的交集 C 就是 P 的凸包[7-10]。

如图 2 所示，在二维平面空间中，凸包就是过集合中的一些点，做凸多边形，使得这个凸多边形能包围所有集合中的点，这个多边形就是凸包。

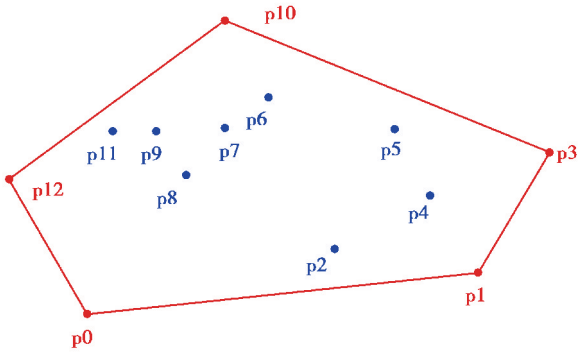

图 2　二维凸包示意

Fig. 2　Two dimensional convex hull diagram

3　Genetic Algorithm

遗传算法(GA，genetic algorithm)，是一种模仿达尔文生物进化论的自然选择原则和遗传学原理的计算模型，是一种寻找最优解的方法。

如图 3 所示，遗传算法首先对要求解的问题值进行编码，然后初始化种群，对种群中的个体进行适应度评估，淘汰适应度不满足的个体，然后从现存的种群中，随机选择一些个体进行交叉和变异，产生新的种群，然后判断是否满足终止条件，如果满足就结束流程，如果不满足则继续进行演化[11-14]。

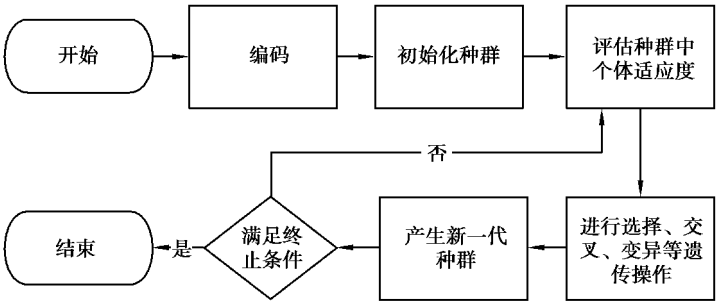

图 3　遗传算法流程图

Fig. 3　Flow chart of genetic algorithm

4 凸包稀疏化及遗传算法参数优化的SVM

针对SVM处理大规模训练数据集存在性能问题和参数优化问题，提出使用凸包算法进行样本数据集稀疏化以及遗传算法进行参数优化的改进SVM算法(CH-GA-SVM)。

4.1 凸包样本稀疏化

由于进行超平面求解所使用的支持向量都在训练数据集的最边缘，所以使用凸包算法进行训练数据集的边缘识别，并使用凸包中的样本点作为训练数据集，这样就对SVM模型训练数据集进行了稀疏化[8-10]。

4.2 遗传算法参数化

SVM参数优化问题主要是对核参数 g 和惩罚因子 c 进行最优值的求解，通常的做法是进行循环验证，所谓循环验证就是把参数 g 和 c 在取值空间中的所有组合进行验证，寻找出最佳的参数组合。这种方法算法的时间复杂度高，需要耗费大量的计算时间。遗传算法参数优化使用遗传算法进行最优 g 和 c 的求解，通过对 g 和 c 进行编码，产生初始种群，然后进行演化迭代，可以快速计算出最优的 g 和 c[11-14]。

4.3 CH-GA-SVM算法流程

如表1所示CH-GA-SVM算法大致分为6个主要步骤：

步骤1：载入原始的训练样本集，并把数据进行归一化处理，对于每个属性值，归一化处理的公式为：$\boldsymbol{V}_{new}=\frac{\boldsymbol{V}_{\text{old}}-\boldsymbol{V}_{\min}}{\boldsymbol{V}^{\max}-\boldsymbol{V}_{\min}}$，其中：$\boldsymbol{V}_{new}$ 为归一后的值；$\boldsymbol{V}_{\text{old}}$ 为归一化前的原值；$\boldsymbol{V}_{\max}$ 为该属性中最大的值；$\boldsymbol{V}_{\min}$ 为该属性中最小的值。

步骤2：把归一化后的样本集作为点集，求解 N 维空间中点集的凸包，维数 N 为输入样本集的属性个数。

步骤3：将凸包求解中返回的样本点进行唯一化处理，使得一个样本点只有一条记录，把唯一化后的凸包点集作为新的训练样本集。

步骤4：定义核参数 g 和惩罚因子 c 的上下界，并进行值的编码，编码采用二级制编码法。

步骤5：初始化染色体基于 g 和 c 参数的种群，开始进行遗传算法迭代，求解最优值。

步骤6：获取遗传算法返回的最优个体，使用最优个体对应的模型进行分类预测，如图4。

表1 CH-GA-SVM算法

Table 1 Algorithm CH-GA-SVM

算法	CH-GA-SVM
输入：	训练数据集 ds_train，测试数据集 ds_test
输出：	分类结果
1：	ds_train = mapminmax(ds_train)；//采用最大最小值归一化方法处理训练数据集
2：	ds_test=mapminmax(ds_test)；//采用最大最小值归一化处理测试数据集
3：	nodes = convhulln(ds_train)；//使用n维凸包算法获取凸包点集
4：	ds_train = unique(nodes)；//采用唯一化方法，过滤凸包点集中重复的点
5：	cB=[−5,5],gB =[−5,5]；//定义c和g的上下边界
6：	maxGen = 10 ;//定义遗传算法进化代数
7：	currentGen = 0 ;//定义当前代数
8：	genSize = 100 ;//定义种群数量
9：	crossoverRate=0.6;//定义交叉概率
10：	pmutateRate=0.001;//定义变异概率
11：	Population = initPopulation(genSize,cB,gB)；//产生初始种群，种群中每个个体的染色体由两个基因组成，一个代表 c，一个代表 g，基因的长度由边界值确定，染色体和基因都是二进制串
12：	For(int i = 0 ; i < genSize ; i++){
13：	[Model,Accuracy]= TrainSVM(Population[i].Chromesone.gene_c.value,

续表1

算法	CH-GA-SVM

```
                Population[i].Chromesone.gene_g.value,ds_train,RBF);//使用种群个体的 c 值和 g 值,训练数据集及 RBF(径向基)核函数,训练 SVM 模型,获取模型的准确率
14:             Population[i].fitness = Accuracy; //使用准确率作为个体的适应度
15:             Population[i].model= Model;//关联训练模型到个体
            }
16:         While(currentGen < maxGen ) {
17:           Population  = Select(Population );//使用赌盘选择法,适应度高的个体被选择概率大,形成新的群体
18:           Population  = Crossover(Population,crossoverRate);//在种群里随机两两配对,然后按预设的交叉概率,两两之间进行染色体随机位置的交叉互换,产生新的种群
19:           Population  = Mutate(Population,pmutateRate);//按照指定的变异概率,在种群中,选择部分个体,进行染色体的变异,随机在染色体的某一位进行 1,0 互换
20:         For(int i = 0 ; i < genSize ; i++){
21:             [Model,Accuracy]= TraimSVM(Population[i].Chromesone.gene_c.value,
                Population[i].Chromesone.gene_g.value,ds_train,RBF);//使用种群个体的 c 值和 g 值,训练数据集及 RBF(径向基)核函数训练 SVM 模型,获取返回的准确率
22:             Population[i].fitness = Accuracy;//使用准确率作为个体的适应度
23:             Population[i].model= Model;//关联训练模型到个体
              }
24:           current Gen ++ ;
            }
25:         Individual = SelectTheBest(Population);//选择适应度最高的个体
26:         Return Individual.model.predict(ds_test); //返回使用个体关联模型对测试数据集进行分类并返回结果
```

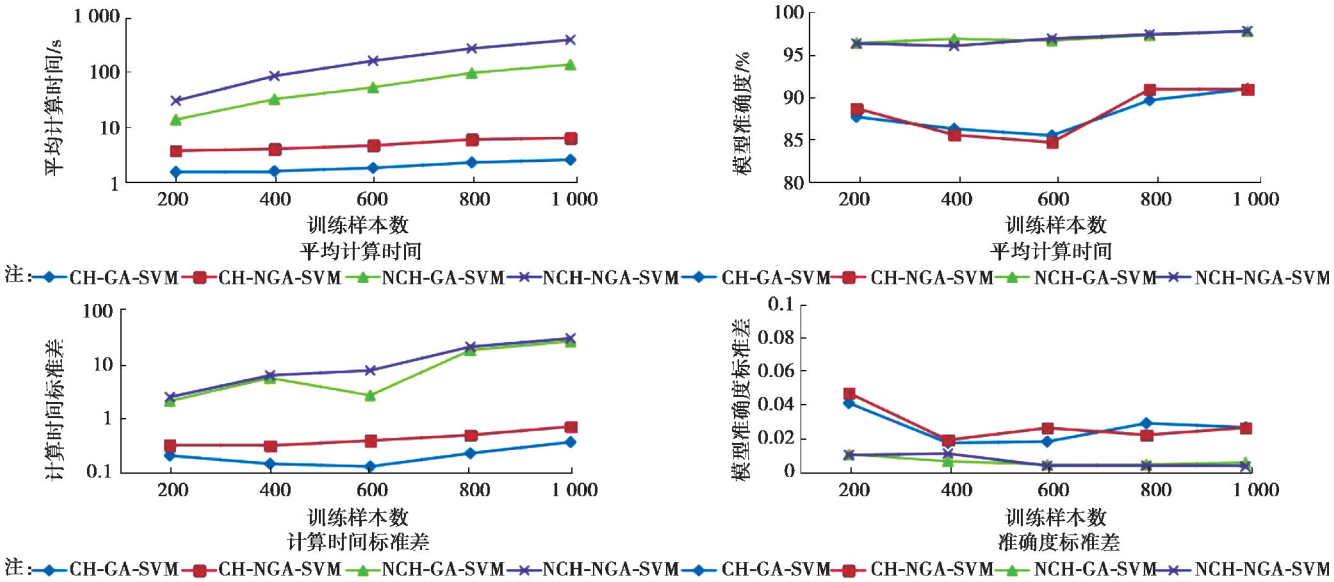

图 4　4 特征数据下各算法对比

Fig. 4　Comparison of algorithms under 4-features data

5 CH-GA-SVM算法实验分析

5.1 实验环境及数据

硬件环境:Intel(R) Core(TM) i7-6700 3.4 GHz CPU及16 GB内存。

软件环境:Windows 10 OS及MATLAB R2019b[15-16]。

实验数据:脱敏后学校贫困生测试数据,包含8个特征,为了进行算法特征数量维度上面的性能分析,取包含其中的4个特征的全量数据作为数据集A,取包含全部的8个特征的全量数据作为数据集B。

数据A:训练数据集1 200条,测试数据集200条,特征数量为4个,分类数为2,贫困和非贫困。

数据B:训练数据集1 200条,测试数据集200条,特征数量为8个,分类数为2,贫困和非贫困。

具体描述如表2~表4所示。

表2 数据集描述

Table 2 Description of datasets

数据集名称	训练数据量	测试数据量	分类数量	数据维度	贫困非贫困数据比例
脱敏贫困生训练数据集A	1 200	200	2	4	5∶9
脱敏贫困生训练数据集B	1 200	200	2	8	5∶9

表3 数据集A示例

Table 3 Example of dataset A

是否贫困生	性别	生源地	月均刷卡次数	月均消费金额/元
0	男	江苏省	261	568
1	女	海南省	156	362

表4 数据集B示例

Table 4 Example of dataset B

是否贫困生	性别	生源地	月均刷卡次数	月均消费金额	月均洗澡次数	月均进入图书馆次数	GPA	月均上网时间/h
0	男	江苏省	261	568	12	15	2.96	144
1	女	海南省	156	362	13	21	3.41	63

5.2 实验对比算法

算法1:NCH-NGA-SVM,未进行训练数据稀疏化,使用循环验证法优化参数的SVM训练算法。

算法2:NCH-GA-SVM,未进行训练数据稀疏化,使用遗传算法优化参数的SVM训练算法。

算法3:CH-NGA-SVM,使用凸包算法进行训练数据稀疏化,使用循环验证法优化参数的SVM训练算法。

算法4:CH-GA-SVM,使用凸包算法进行训练数据集稀疏化,使用遗传算法优化参数的SVM训练算法。

5.3 实验指标及方法

使用训练SVM模型消耗的平均计算时间、方差和模型的分类平均准确率、方差作为实验的指标。在相同数量的样本的对比实验中,采用5-折交叉验证法,分5次替换数据,每次替换样本中20%的数据,然后每次

替换后样本测试 3 次，取总共 15 次的平均值和方差作为最后实验结果。

5.4　实验结果与分析

表 5　实验结果数据

Table 5　Experimental data

数据集		NCH-NGA-SVM				NCH-GA-SVM				CH-NGA-SVM				CH-GA-SVM			
		Time AVG	Time STD	Acc AVG	Acc STD	Time AVG	Time STD	Acc AVG	Acc STD	Time AVG	Time STD	Acc AVG	Acc STD	Time AVG	Time STD	Acc AVG	Acc STD
数据集A	200 样本	16.51	0.41	0.97	0.010	10.04	1.86	0.97	0.011	2.20	0.12	0.89	0.048	1.55	0.23	0.88	0.042
	400 样本	52.81	0.68	0.96	0.011	28.32	5.48	0.97	0.007	2.45	0.18	0.86	0.019	1.61	0.16	0.86	0.018
	600 样本	107.52	5.13	0.97	0.004	48.28	2.39	0.97	0.004	2.84	0.28	0.85	0.027	1.84	0.15	0.85	0.018
	800 样本	171.06	2.90	0.98	0.004	90.94	18.01	0.98	0.005	3.71	0.29	0.91	0.022	2.30	0.25	0.90	0.030
	1 000 样本	247.86	3.78	0.98	0.003	131.27	25.94	0.98	0.006	3.83	0.37	0.91	0.027	2.58	0.40	0.91	0.027
数据集B	200 样本	16.38	0.41	0.97	0.010	9.86	2.04	0.97	0.010	2.27	0.12	0.89	0.048	1.46	0.13	0.89	0.046
	400 样本	53.29	0.67	0.96	0.011	26.12	4.46	0.97	0.005	2.51	0.17	0.86	0.019	1.58	0.16	0.86	0.017
	600 样本	106.34	0.90	0.97	0.004	48.31	2.70	0.97	0.005	2.87	0.25	0.85	0.027	1.81	0.17	0.86	0.017
	800 样本	170.25	1.74	0.98	0.004	84.36	20.08	0.98	0.003	3.73	0.33	0.91	0.022	2.27	0.25	0.90	0.019
	1 000 样本	248.83	1.89	0.98	0.003	130.26	14.76	0.98	0.004	3.86	0.40	0.91	0.027	2.51	0.24	0.91	0.027

在模型训练计算时间上，由于各个算法计算时间差别很大，计算平均时间图和方差图纵坐标采用对数坐标。如图 4、5 所示，在不同的样本数量和特征数量下面，传统的 NCH-NGA-SVM 算法模型计算时间最长，通过单独采用凸包或者遗传算法后，计算的时间都明显减少，凸包和遗传算法都使用的 CH-GA-SVM 模型计算的时间最短，效率最高。

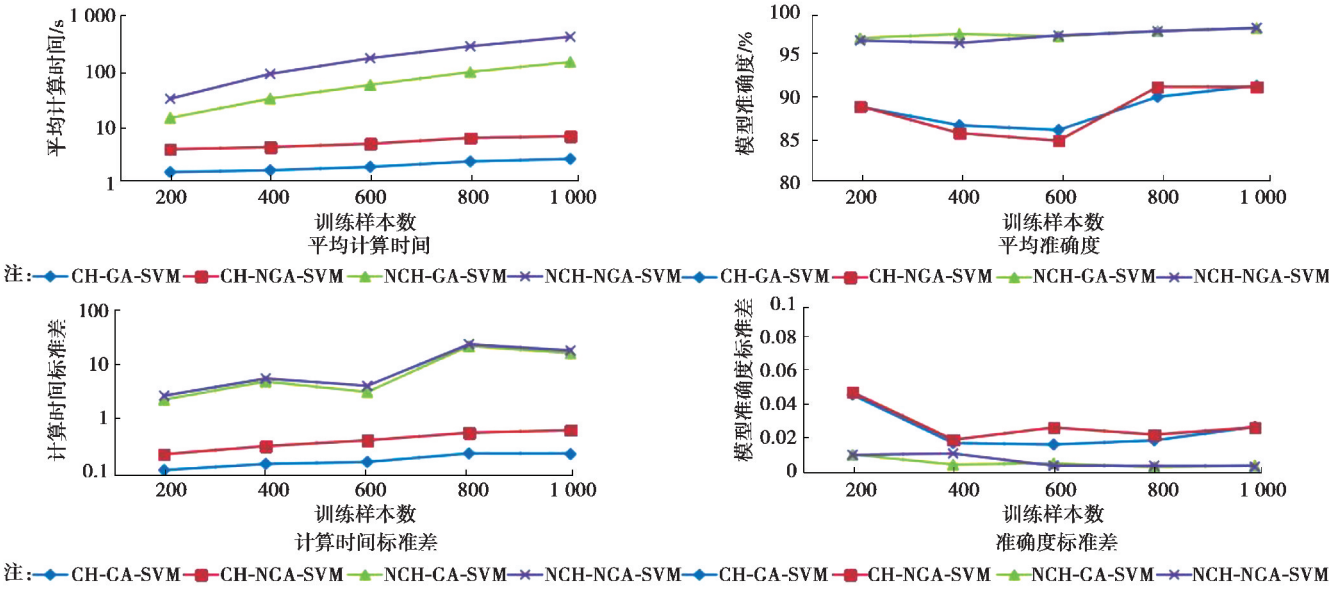

图 5　8 特征数据下各算法对比

Fig. 5　Comparison of algorithms under 8-features data

在模型准确率方面，在样本的较少的训练数据下，采用凸包算法的 CH-GA-SVM 和 CH-NGA-SVM 准确率相对较低，这是因为凸包算法对数据做了稀疏化，输入的训练数据有失真造成的，但是当训练样本逐渐

增多后，CH-GA-SVM算法的准确率也逐步提高，在1 000训练样本数据下准确率可以达到90%以上。同时随着训练样本数据的增加，模型准确率的标准差，也越来越小，表明模型稳定性越来越高。不同的特征数量下，特征越多各算法的计算时间都会相应增加，准确率方面差别不大。

6 结 论

笔者提出的CH-GA-SVM算法，通过凸包算法进行样本稀疏化，使用遗传算法进行核参数 g 和惩罚因子 c 优化，确实能显著改善SVM训练效率，缩短计算时间。同时凸包算法进行样本稀疏化后，模型准确度仍然能维持在一个较高水平，证明凸包算法稀疏化的样本集能很大程度保留原始数据的特征，使用凸包算法进行SVM训练样本稀疏化，具有一定的可行性。但是进行凸包稀疏化后，造成训练数据的部分失真，模型准确度下降，如何进行算法上的进一步改进，解决这个问题，后续还需要进行专门的研究。

参考文献：

[1] Imbault F, Lebart K. A stochastic optimization approach for parameter tuning of Support vector machines[C]// Proceeding of the 17th International Conferenceon Pattern Recognition. Cam bridge, United Kingdom:[s.n.], 2004: 981-984.

[2] Han J W, Micheline K B, Pei J. Data mining: concepts and techniques (third edition)[M]. San Francisco: Morgan Kauf mann, 2011: 211-286.

[3] De Brabanter K, De Brabanter J, Suykens J A K, et al. Optimized fixed-size kernel models for large data sets[J]. Computational Statistics & Data Analysis, 2010, 54(6): 1484-1504.

[4] Huang K Z, Zheng D N, Sun J, et al. Sparse learning for support vector classification[J]. Pattern Recognition Letters, 2010, 31(13): 1944-1951.

[5] Theodorids S, Koutroumbas K. Pattern recognition, third edition[EB/OL]. 2006. https://www.researchgate.net/publication/258023950_Pattern_Recognition_Third_Edition.

[6] Bao T Q, Mordukhovich B S. Relative pareto minimizers for multiobjective problems: existence and optimality conditions [J]. Mathematical Programming, 2010, 122(2): 301-347.

[7] 周培德. 计算几何—算法分析与设计[M].北京:清华大学出版社,2003.
Zhou P D. Computational geometry-algorithm analysis and design[M]. Beijing: Qinghua University Press, 2003.

[8] Ding S G, Nie X L, Qiao H, et al. A fast algorithm of convex hull vertices selection for online classification[J]. IEEE Transactions on Neural Networks and Learning Systems, 2018,29(4): 792-806.

[9] Gu X Q, Chung F L, Wang S T. Fast convex-hull vector machine for training on large-scale ncRNA data classification tasks[J]. Knowledge-Based Systems, 2018, 151: 149-164.

[10] Wang D, Qiao H, Zhang B, et al. Online support vector machine based on convex hull vertices selection[J]. IEEE Transactions on Neural Networks and Learning Systems, 2013, 24(4): 593-609.

[11] Royachka K, Karova M. High-performance optimization of genetic algorithms[C]// 2006 29th International Spring Seminar on Electronics Technology, May 10-14, 2006. Germany: IEEE, 2006.

[12] Srinivas M, Patnail L M. Adaptive probabilities of crossover and mutation in genetic algorithms[J]. IEEE Transactions on Systems, Man and Cybernetics, 1994, 24(4): 656-667.

[13] Chuang Y C, Chen C T, Hwang C. A simple and efficient real-coded genetic algorithm for constrained optimization [J]. Applied Soft Computing, 2016, 38: 87-105.

[14] Michalewicz Z. Evolution programs and heuristics[M]. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996: 307-327.

[15] Chang C C, Lin C J. "LIBSVM: A library for support vector machines[J]. ACM Transactions on Intelligent Systems & Technology, 2011.2(3): 389-396.

[16] Gong C, Wang Z L. Proficient optimization with mailab[M]. Beijing: Publishing House uf Electronics Industry, 2009: 315-320.

（编辑 侯 湘）