第34卷 第17期 计 算 机 工 程 2008年9月
Vol.34 No.17 Computer Engineering September 2008
·博士论文· 文章编号：1000—3428(2008)17—0028—03 文献标识码：A 中图分类号：TP312

一种基于凸壳算法的SVM集成方法

张宏达，王晓丹，白冬婴，刘惊源
(空军工程大学导弹学院，三原 713800)

摘 要：为提高支持向量机(SVM)集成的训练速度，提出一种基于凸壳算法的SVM集成方法，得到训练集各类数据的壳向量，将其作为基分类器的训练集，并采用Bagging策略集成各个SVM。在训练过程中，通过抛弃性能较差的基分类器，进一步提高集成分类精度。将该方法用于3组数据，实验结果表明，SVM集成的训练和分类速度平均分别提高了266%和25%。

关键词：凸壳算法；支持向量机；集成

SVM Ensemble Approach Based on Convex-hull Algorithm

ZHANG Hong-da, WANG Xiao-dan, BAI Dong-ying, LIU Jing-yuan
(Missile Institute, Air Force Engineering University, Sanyuan 713800)

【Abstract】To improve the training speed of Support Vector Machine(SVM) ensemble, this paper proposes a new approach of SVM ensemble using convex-hull algorithm. The approach applies convex-hull algorithm to get from each class the hull vectors and takes these hull vectors as the training dataset for every base-classifier, Bagging method is used to aggregate the base-classifiers. Threshold is set to discard the base-classifiers with weak performance in training the ensemble to further improve the classification accuracy. Experimental results obtained from applying the proposed approach to 3 different datasets indicate that on average it accelerates training by 266% and speeds up classifying by 25%.

【Key words】 convex-hull algorithm; Support Vector Machine(SVM); ensemble

1 概述

集成学习是当前机器学习四大研究方向之一[1]。支持向量机集成(Support Vector Machine Ensemble, SVME)是指按照一定规则将有限个子SVM的结果结合起来，以便对新样本进行分类预测的学习算法。通过SVM集成，可以在一定程度上避免SVM本身的模型选择问题，并可能获得比单个SVM更好的泛化性能。通常，每个SVM的训练过程表现为一个解决凸二次优化的问题，在解决大规模数据集问题时，受到运算时间和存储空间的考验。因为利用凸二次优化技术解决问题时，需要进行大量的矩阵运算，耗费大量运算时间，而且优化过程中需要存储一个核矩阵，存储空间随着训练数据集规模的增大呈平方增长，所以对于SVM集成，需要解决的一个重要问题是在保证分类精度的前提下，提高基分类器即各个SVM的训练速度。为提高SVM的训练速度，许多方法分别从不同方面对SVM的训练过程进行了优化和改进[2-4]，SMO, chunking方法等分解算法[2]通过解决多个小规模QP问题逐步逼近最优解，从而避免存储整个Hessian矩阵，成为目前解决大规模SVM训练的主要方法；基于数据分布的密度、距离特征，通过聚类进行的工作集约简[3-4]是缩小训练集、提高训练速度的另一有效途径。

SVM的本质是在2类数据之间求取最优分类超平面，从训练样本的几何分布角度看，最有可能成为支持向量的样本分布在每类数据的凸壳上或者靠近凸壳的区域。凸壳向量往往只占数据集的很小一部分，使用壳向量作为新的训练集可以提高SVM增量学习速度[5]。为提高SVM集成的训练速度，本文将壳向量引入到基分类器SVM的训练过程中，并采用Bagging集成，从而提出了一种基于凸壳算法的SVM集成方法(记为HBag)。

2 凸壳算法简介

构造SVM分类器时，训练集中往往只有一小部分点可能成为支持向量。若能从训练样本中选择出最有可能成为支持向量的样本，并只对这些样本进行训练，将大大减小训练集规模，在保证分类精度的同时，明显提高训练速度。

壳向量是指所有位于训练集最边缘的样本，即位于训练集凸壳上的样本。事实上，壳向量就是样本集的凸顶点。研究表明[5]，对于线性可分情形，支持向量集是壳向量的子集。

根据计算几何理论[6]：

(1) $n\to\infty$，k 维球体中均匀独立随机分布 n 个点，壳向量个数 $m(k)=O(n^{(k-1)/(k+1)})$；

(2) $n\to\infty$，若数据点呈 k 维正态分布，壳向量个数 $m(k)=O((\ln n)^{(k-1)/2})$；

(3) k 维空间 n 个点的分量是独立地从任何连续分布的集合中随机选取的，其壳向量个数 $m(k)=O\left((\ln n)^{k-1}\right)$。

对任意一种情况，均有 $\lim\limits_{n\to\infty}\dfrac{m(k)}{n}=0$，上述分析表明壳向量个数远远小于原数据点集的个数。

根据计算几何理论中的相关定理，计算数据集 S 的壳向量集合(凸壳)的算法时间复杂度至少为 $O(n\ln n)$。

事实上，凸壳算法对于数据集是有限制的。本文在实验

基金项目：国家自然科学基金资助项目(50505051)；陕西省自然科学基金资助项目(2007F19)；空军工程大学导弹学院研究生学位论文创新基金资助项目(DY06102)

作者简介：张宏达(1981－)，男，博士研究生，主研方向：智能信息处理，机器学习；王晓丹，教授、博士；白冬婴、刘惊源，硕士研究生

收稿日期：2007-09-15 **E-mail：**zhdhonda@163.com

部分对求凸壳算法的效率与适用范围进行了分析。对于高维数据，凸壳算法的耗时将随着样本集的增加急剧增长，甚至远超过直接用全体训练集训练 SVM 的时耗，这种情况下可以通过一些降维方法，如主成分分析(PCA)或特征选择方法，对数据进行预处理，得到适用于凸壳算法的数据集。

3 基于凸壳算法的 SVM Bagging 集成

弱可学习表示学习算法识别一组概念的正确率仅仅略好于随机猜测，即正确率略大于 50%。1990 年 Schapire 证明了弱学习算法和强学习算法是等价的。在学习时，只需要找到比随机猜测略好的弱学习算法，就可以通过集成将其提升为强学习算法，而不必直接寻找强学习算法，因为强学习算法在通常情况下很难获得[7]。

Bagging 集成[8]是一种常用的集成策略，它基于自举采样(bootstrap sample)策略，每次随机从训练样本集中抽取一个子集作为训练样本集，利用训练样本子集构造出一个基 SVM，最后通过对这些 SVM 进行投票决策得出分类结果。采用凸壳算法可进一步减少基分类器的训练集及支持向量个数，从而提高集成的训练速度和分类速度。

基分类器之间的差异性是通过集成提高分类精度的必要条件。随机采样时，有时采样集的分布与原数据集的分布差异过大，导致对应基分类器的分类误差过高，甚至分类结果不及随机猜测，这种基分类器的加入显然不会带来集成性能的提高，因此，本文通过设置阈值对这种基分类器作抛弃处理。虽然理论上，只要基分类器的分类正确率略大于 50%，就可以通过某种策略得到分类正确率很高的集成分类器，但是要用略好于随机猜测的基分类器得到一个强分类器，需要大量的基分类器，会严重影响集成算法收敛速度；若只保留分类正确率很高的基分类器，又将损失基分类器之间的差异度。综合考虑以上 2 点，不同的分类问题应设定不同的抛弃阈值 $err_{\text{threshold}}$，建议该值设得比基分类器分类误差的均值稍高。

对于一个二类分类问题，已知有标记样本集 $L=\{(x_i,y_i),i=1,2,\cdots,n\}$，本文基于凸壳算法的 SVM Bagging 集成(HBag)方法步骤如下：

(1)训练集 L，抛弃阈值 $err_{\text{threshold}}$，基分类器个数 T，i=1；

(2)采样 $Ls_i=\text{bootstrap samples from } L$；

(3) $L_i=\text{convHulln}(Ls_i)$；

(4)训练第 i 个基分类器 $j_i=\text{Learner}(L_i)$；

(5)用训练集 L 计算误分率 $err(j_i)$，若 $err(\varphi_i)>err_{\text{threshold}}$，抛弃 φ_i，转到(2)；否则，i=i+1，转到(2)；

(6)对测试样本分类

$$\varphi^*(x)=\text{sign}(\sum_{i=1}^{T}\varphi_i(x))$$

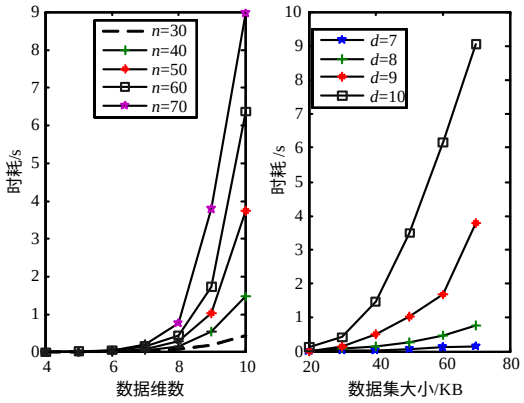
4 实验与分析

首先对凸壳算法的时耗进行实验，分析了该算法的适用数据集范围；其次通过多组数据将 HBag 算法与单个 SVM，Bagging SVM 进行了精度、训练时间对比，检验该算法的性能。实验机器配置为内存 256 MB，Sempron 3000+(1.80 GHz)，运行环境为 Windows XP，Matlab7.0，采用 Steve Gunn 的 SVM 工具箱。

4.1 凸壳算法的时间耗费

采用 Matlab 工具箱中 convhulln 函数计算凸壳，其中，convhulln 函数由 Quick hull 算法实现。

对于高维数据，采用 UCI 中 Wine 数据集，对其进行凸壳计算的时间耗费如图 1 所示。

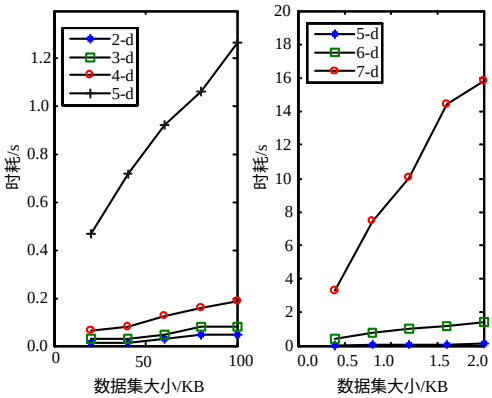


(a)数据维数　　(b)数据集

图 1　高维数据求凸壳时耗

图 1(a)示出了不同规模的样本集，计算凸壳耗费时间随样本维数增加的曲线；图 1(b)示出了不同维数的样本，计算凸壳耗费时间随样本数目增长的曲线。可见，当数据集维数增大时，计算凸壳的耗时将剧增，例如，对于规模为 70 的数据集，当维数从 7 维增加到 10 维时，耗时从 0.172 s 剧增到 8.95 s；计算凸壳的耗时随数据集规模增大也迅速增大，例如，取数据维数为 9 维，当数据集规模从 40 增加到 70 时，耗时从 0.516 s 增加到 3.766 s。同时，求凸壳的耗时随着维数增加而增加远比随数据集规模的增大而增加的速度更快。

对于中、低维数据，图 2(a)的数据为 2 维~5 维，数据集中每一分量均服从 Gauss 分布，数据集大小从 20 000~100 000 时求凸壳算法耗时；图 2(b)所示为数据维数为 5, 6, 7，数据集大小从 400~2 000 时算法的耗时。可见，对于低维数据，即使大数据集求凸壳算法仍然有较好的性能。而当维数由 6 到 7 时，中型数据集求凸壳所需时耗急剧增加。因此，对于一般应用而言，凸壳算法适用于 2 维~7 维情形。



(a)数据维数　　(b)数据集

图 2　中、低维大中型数据集求凸壳时耗

由上述分析可知，凸壳算法适用范围为：(1)低维大、中、小规模数据集(维数≤5)；(2)较高维中、小规模数据集(6≤维数≤8)；因此若要体现凸壳算法的优势，对于高维数据，可通过选取特征子集以及维数约减的方法得到较低维的数据；对于大数据集，可通过采样得到一个较小的数据子集。

4.2 HBag 的性能检验

共采用 3 组数据集对提出的方法进行验证，其中 2 组来

源于 UCI 数据集，1 组为仿真数据。

4.2.1 实验数据设置

(1)UCI 的 Balance 数据库，共 627 组数据，类标为 1 的作为正类，类标为 2, 3 的数据作为负类，前 300 组为训练集，后 327 组为测试集。

(2)人工数据集 Art1，采用服从 4 维 Gauss 分布的二类别仿真数据，2 类中心欧氏距离为 4，正例 500 例，负例 500 例，前 300 组为训练集，后 700 组为测试集。

(3)UCI 的 Hepatitis 数据库，经缺失属性值预处理并应用 PCA 方法将输入数据维数降为 4 维，最终输入数据为 154 组，32 组正例，122 组反例，前 100 组作为训练集，后 54 组为测试集。

训练 SVM 时采用 RBF 核，对 3 组数据，依次设置 $\sigma=2$，$\sigma=3$，$\sigma=1$，参数 C 都为 100。

4.2.2 实验结果

分别采用单个 SVM、基于壳向量的 SVM(HSVM)、Bagging SVM(Bag)、基于壳向量的 Bagging 集成(HBag)4 种方法对 2 个数据集进行了实验；在实验中，集成规模为 T=31, T=23, T=11 时，表 1、表 2 中分别用 Bag(1), Bag(2), Bag(3) 以及 HBag(1), HBag(2), HBag(3)表示，各实验 10 次取平均值得到表 1、表 2，其中，*Ac* 表示精度，*Tt* 表示训练时间。

表 1 单个 SVM, Bagging SVM 集成和新方法的性能对比

	Balance		Art1		Hepatitis	
	Ac	*Tt*	*Ac*	*Tt*	*Ac*	*Tt*
SVM	0.954	33.5	0.924	29.3	0.778	1.6
HSVM	0.840	0.3	0.930	1.1	0.648	0.4
Bag(1)	0.955	116.6	0.968	116.1	0.826	20.6
Bag(2)	0.953	85.4	0.965	86.3	0.838	12.9
Bag(3)	0.951	41.7	0.962	33.7	0.794	6.8
HBag(1)	0.957	28.1	0.966	31.1	0.760	9.1
HBag(2)	0.953	24.0	0.967	21.9	0.750	6.6
HBag(3)	0.942	11.3	0.963	6.82	0.749	3.1

表 2 Bag 和 HBag 的支持向量总数对比

	Bag (1) / (2) / (3)	HBag(1) / (2) / (3)
Balance	850 / 636 / 305	657 / 490 / 231
Art1	358 / 264 / 130	294 / 233 / 112
Hepatitis	1 203 / 885 / 424	978 / 724 / 346

从表 1 可以看出，采用 HSVM 的训练时间比 SVM 大大缩减，但是分类精度依赖于数据集分布，对于 Balance 和 Hepatitis 数据集，其分类精度均有所下降。对于前 2 个数据集，采用 Bag 和 HBag 均能提高分类精度，但是相比传统 Bag 采用 HBag 能大幅减少训练时间。如表 1 所示，对于前 2 个数据集，采用 HBag 训练耗时小于单个 SVM 或与单个 SVM 相当，却能获得更高的分类精度。

如表 2 所示，HBag 支持向量总数比 Bag 减少了 10%~20%，比采用 Bag 分类决策速度更快。

同时从表 1 可以看出，集成分类精度并不随着基分类器个数的增加而提高，这就牵涉到选择性集成[9]的问题，如何从训练的基分类器中选择一部分参与集成使得集成最优化，是集成学习中又一重要课题。

在实验中，对于前面 2 个数据集，训练出的基分类器性能一般都远好于随机猜测，因此，抛弃阈值并没有起作用。对于第 3 个数据集，由于 PCA 降维以后，数据集中不同类别之间的交叠严重，因此 2 类数据的凸壳之间严重重叠，进而导致基于凸壳的集成性能的退化；用 HBag 方法时，基分类器的分类精度在 0.5~0.7 之间，未使用抛弃处理时发现，集成分类性能不稳定。实验中设定 $err_{\text{threshold}}$=0.6，大大提高了集成分类精度和稳定性，同时因为有约 50%的基分类器被抛弃，使得训练时间增加近 1 倍；尽管如此，对第 3 组数据集 HBag 方法，性能仍然比 Bag 差很多，甚至不如单个 SVM。

5 结束语

本文提出了一种基于凸壳算法的 SVM Bagging 集成方法，该方法中 Bagging 的随机采样技术可以保证靠近凸壳的点有机会进入工作集，从而可以充分利用数据集的信息，对于不同类别之间数据交叠不严重的数据集，该方法能大幅提高集成训练及分类速度。如何使该方法更有效地处理高维线性不可分问题，还需要进一步的研究。

参考文献

[1] Dietterich T G. Machine Learning Research: Four Current Directions[J]. AI Magazine, 1997, 18(4): 97-136.

[2] 李建民, 张 钹, 林福宗. 支持向量机的训练算法[J]. 清华大学学报: 自然科学版, 2003, 43(1): 120-124.

[3] 李晓黎, 刘继敏, 史忠植. 基于支持向量机与无监督聚类相结合的中文网页分类器[J]. 计算机学报, 2001, 24(1): 62-68.

[4] 武方方, 赵银亮. 一种基于 Morlet 小波核的约简支持向量机[J]. 控制与决策, 2006, 21(8): 848-852.

[5] 李东晖, 杜树新, 吴铁军. 基于壳向量的线性支持向量机快速增量学习算法[J]. 浙江大学学报: 工学版, 2006, 40(2): 203-207.

[6] 普雷帕拉塔, 沙莫斯. 计算几何导论[M]. 庄心谷, 译. 北京: 科学出版社, 1990: 183-187.

[7] 王 钰, 周志华, 周傲英. 机器学习及其应用[M]. 北京: 清华大学出版社, 2006: 13-14.

[8] Breiman L. Bagging Predictors[J]. Machine Learning, 1996, 26(2): 123-140.

[9] Zhou Zhihua, Wu Jianxin, Tang Wei. Ensembling Neural Networks: Many Could Be Better than All[J]. Artificial Intelligence, 2002, 137(1/2): 239-263.

（上接第 27 页）

参考文献

[1] Daganzo C F. Requiem for Second-order Fluid Approximations of Traffic Flow[J]. Transportation Research, 1995, 29(4): 277-286.

[2] Lebacque J P, Lesort J B. Macroscopic Traffic Flow Models: A Question of Order[C]//Proceedings of the 14th International Symposium on Transportation and Traffic Theory. Jerusalem, Israel: [s. n.], 1999: 3-25.

[3] Papageorgiou M. Modeling and Real-time Control of Traffic Flow on the Southern Part of Boulevard Peripherique in Paris: PART1 Modeling[J]. Transportation Research, 1990, 24(5): 345-359.

[4] Wang Yibing, Papageorgiou M. Renaissance—A Unified Macroscopic Model-based Approach to Real-time Freeway Network Traffic Surveillance[J]. Transportation Research Part C, 2006, 14(3): 190-212.

[5] Kalman R E. A New Approach to Linear Filtering and Prediction Problems[J]. Trans. of the ASME J. of Basic Engineering, 1960, 82(1): 34-35.

[6] Xu Tiandong, Hao Yuan, Sun Lijun. A New Travel Time Prediction of Urban Expressway in Unstable Traffic Flow[C]//Proc. of International Conference on Transportation Engineering. Shanghai, China: [s. n], 2205-2210.