第35卷 第24期 计 算 机 工 程 2009年12月
Vol.35 No.24 Computer Engineering December 2009
·人工智能及识别技术· 文章编号：1000—3428(2009)24—0188—03 文献标识码：A 中图分类号：TP301.6

基于相邻边界模型的支持向量预选算法

孙 卫[1]，庄卫华[2]，林红飞[2]，曾晓勤[2]
（1. 南京审计学院信息科学学院，南京 210029；2. 河海大学计算机与信息工程学院，南京 210098）

摘 要：根据支持向量的几何分布特性，提出相邻边界模型的概念以及一种支持向量预选算法。该算法通过预选出相互邻近的边界样本，避免大量样本参与二次规划问题的求解，为支持向量机提供高效的训练集。实验结果证明，采用该预选算法的LIBSVM可以较大地提高训练的时间效率和空间效率。

关键词：相邻边界模型；支持向量机；主动学习

SV Pre-selecting Algorithm Based on Adjacent Boundary Model

SUN Wei[1], ZHUANG Wei-hua[2], LIN Hong-fei[2], ZENG Xiao-qin[2]
（1. School of Information Science, Nanjing Audit University, Nanjing 210029;
2. College of Computer and Information Engineering, Hohai University, Nanjing 210098）

【Abstract】 According to the geometry distribution property of Support Vector(SV), this paper proposes the concept of adjacent boundary model and SV pre-selecting algorithm. By pre-selecting adjacent boundary samples, lots of samples are avoided to solve Quadratic Programming(QP) problems, which provides efficient training sets for SVM. Experiments show that it can improve efficiencies of training time and space by using the algorithm in LIBSVM.

【Key words】 adjacent boundary model; Support Vector Machine(SVM); active learning

1 概述

支持向量机(Support Vector Machine, SVM)[1]以严格的数学理论为基础，具有简洁的数学形式、直观的几何解释和良好的泛化能力，与核技术的结合使它成为解决分类、回归、概率密度估计等实际问题的有力工具。但SVM问题的本质是求解规划问题，因此，求解过程比较漫长，特别是当Hessian矩阵为奇异矩阵或有很小的特征值时，求解过程尤为缓慢。另外由于需要存储Hessian矩阵以及其他中间变量，因此需要较大的存储空间，当训练样本较多时，这一问题会变得更加突出。为此，许多学者从各种角度提出了SVM的快速算法，例如：减小每次优化问题的规模，通过多次优化寻找最优解，代表性的算法有Osuna, Smo等；变换求解问题或二次规划问题，典型的算法有CVM, Least Squares SVM等。它们在一定程度上提高了SVM的求解速度、降低了对存储空间的要求，但时间和空间效率仍是限制SVM应用的瓶颈。

在两分类问题中，SVM的超平面表达式如下：

$$Y(x)=\omega X+b,\ \omega\in R^N,\ b\in R,\ \omega=\sum_{i=1}^{m}a_i y_i x_i \tag{1}$$

其中，ω由a_i不为0所对应的样本点决定，这些样本点称为支持向量(Support Vector, SV)，a_i的大小决定了该点对超平面所做的贡献。大多数情况下，支持向量只占训练样本很少的一部分并且位于2类样本的相邻边界处，其余大多数样本对SVM的分类超平面没有贡献，因此，在训练SVM的求解过程中，许多存储空间和运算被浪费了。如果能在训练SVM前预选出这些可能的支持向量，就能在不影响准确性的情况下大大减少训练时间和存储空间。

目前这方面的研究包括使用两凸包相对边界向量方法预抽取支持向量[2]、使用中心比值法预选支持向量[3]、使用自适应投影算法预选支持向量[4]等。本文给出了在某种条件下算法复杂度为线性的支持向量主动学习策略，并使用核函数把该方法推广到特征空间，从而解决了非线性可分问题的支持向量预选问题。

2 基于相邻边界模型的支持向量预选算法

2.1 相邻边界模型

定义 训练集T

$$T=\{T^+\cup T^-\}=\{(x_1,y_1),(x_2,y_2),\cdots,(x_M,y_M)\}\in(X\times Y)^M$$

其中，T^+为正类点集；T^-为负类点集；$x_i\in X=R^n$，$y_i\in Y=\{1,-1\}$，$i=1,2,\cdots,M$。设$d(x_1,x_2)=\min\{d(x_1,x_j)\}$，其中，$x_1\in T^+$，$x_2,x_j\in T^-$，则称$x_2$为$T^+$的相邻边界点。

可以采用贪婪的策略求解相邻边界点，但该方法比较费时。本文采用交叉求解的策略，即通过A类中的一点在另一类B中求出A的相邻边界点，接着以该相邻边界点为基础，求出B的相邻边界点，如此往复。一开始，任取一点为起始点。

图1给出了线性可分问题中相邻边界点的例子，其中，方块表示正类；空心圆表示负类；箭头所指的样本点为相邻边界点；点A为噪音点，分布在负类的内部。可以看出，大部分相邻边界点分布在各自所属类别的边界上，B点除外，因为A点为其所属集合的噪音点。

基金项目：国家自然科学基金资助项目(60571048)
作者简介：孙 卫(1971－)，女，硕士，主研方向：数据挖掘，网络安全；庄卫华，副教授；林红飞，硕士；曾晓勤，教授、博士生导师
收稿日期：2009-09-11 **E-mail：**sww@nau.edu.cn

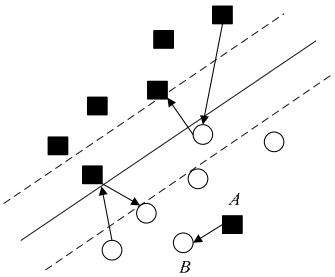

图 1　相邻边界点在线性可分问题中的分布示例

2.2　支持向量预选算法

由空间几何性质可知，对于线性可分问题，相邻边界点一般分布在点集的边界处，因此，要求出支持向量只需求出相邻边界点。在求相邻边界点之前为训练集中的每个样本点设置一个标记位，并规定标记 0 表示没有访问过该点，1 表示访问过该点，2 表示不但访问过该点而且该点为相邻边界点，初始时设所有样本点的标记为 0。预选算法流程见图 2，其中，变量 x_1, x_2 的意义同定义；变量 r 用于记录连续设置样本标记为 2 的次数，当 r 大于某一阈值 R 时，可以认为已求出所有的相邻边界点；取距离 x_1 最远的点是为了避免因局部区域而使 r 值连续增大，防止预选过程过早收敛。

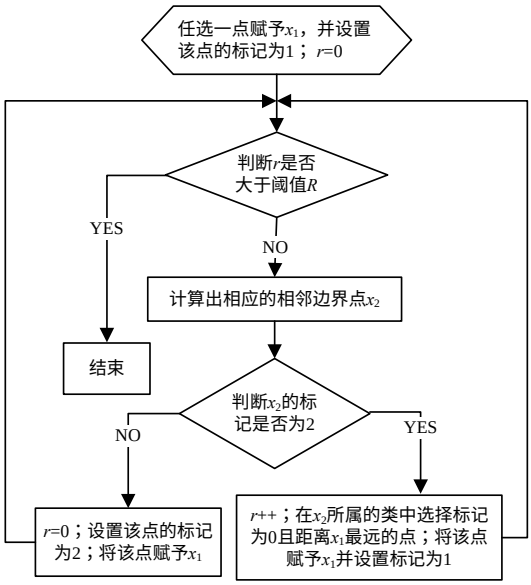

图 2　基于相邻边界模型的支持向量预选流程

少量噪音点的存在可能使小部分相邻边界点分布在样本集的内部，但这并不影响最终训练集的质量，反而因它们的加入增加了训练集的信息，提高了训练的准确性。图 2 中阈值 R 的选取直接影响算法的效率，取值太小，预选过早地收敛，不能求得所需的相邻边界点，取值太大，不但会增加时间开销，而且无谓地增加了标记为 1 的样本点。针对这个问题，本文事先采用逐步递增的方法来确定阈值 R 的大小，直至标记为 2 的样本点个数不再变化时停止递增。

对于非线性可分问题，运用核技术将不可分样本点集映射到高维特征空间中，使这些样本点在高维 Hilbert 空间中线性可分，然后采用如上的预选策略求出所需的相邻边界点。特征空间中 2 个样本点之间距离计算公式如下，其中，Ψ 为满足 Mercer 条件的非线性变换；K 是对应的核函数：

$$d(\Psi(x_1),\Psi(x_2))=\sqrt{(\Psi(x_1)-\Psi(x_2))\cdot(\Psi(x_1)-\Psi(x_2))}=\sqrt{K(x_1,x_1)-2K(x_1,x_2)+K(x_2,x_2)} \quad (2)$$

2.3　算法复杂度分析

设 M 为训练集大小，正负类样本各占一半，n 为采样后训练集的大小。通过分析上面的主动学习策略可知，该采样过程最好情况是一次性连续预选出相邻边界点，这时的采样时间复杂度为 $O(\frac{1}{2}nM)$；最坏情况是每求出一个标记为 2 的样本都要 R 次循环，其中，R 为预先定义的阈值，这时的时间复杂度为 $O(\frac{1}{2}RnM)$。以求解 QP(Quadratic Programming)问题的时间复杂度一般情况下为 $O(N^2)$ 计算，则采用本主动学习策略的 SVM 求出最终超平面的时间复杂度在最好情况下为 $O(\frac{1}{2}nM+n^2)$，在最坏情况下为 $O(\frac{1}{2}nRM+n^2)$。在采样比例很大的情况下，即 $n<<M$ 时，时间复杂度接近线性，变为 $O(M)$，并且训练时只需占用很少的内存。

3　实验和分析

为了验证本文预选算法的有效性和实用性，将 SVM 的开源软件包 LIBSVM 作为实验平台。实验的方法是在 LIBSVM 原程序训练模块的开始部分插入用 C 语言实现的预选代码，用于提取相邻边界点，为后期的训练提供数据源。实验的目的是通过比较在没有添加该预选模块之前和添加之后 LIBSVM 的准确性和时空效率，证明该预选模型的有效性和高效性。R 取 20，因为通过对多个数据集进行实验后发现，该值能提取所有或绝大部分相邻边界点。训练选用的支持向量机为 C-SVM，用高斯函数 $\exp(-gama||Y-X||^2)$作为核映射函数，其中，参数 C 和 $gama$ 的取值通过交叉验证决定。共进行 2 个实验，第 1 个实验是为了验证本采样策略的有效性和高效性；第 2 个实验是为了进一步验证它的高效性及实用性。

3.1　实验一

用 LIBSVM 主页上提供的 2 类数据作为实验数据源，共进行 3 组实验，每组使用共同的参数值，以便于比较。数据源的部分信息和实验时的参数取值如表 1 所示。

表 1　实验数据信息及训练参数取值

训练集及大小	*Features*	测试集及大小	*C*	*gama*
ijcnn1, 49 996	22	ijcnn1.t, 91 701	10	1.00
a1a, 1 606	123	a1a.t, 30 956	10	0.01
a9a, 32 561	123	a9a., 16 281t	10	0.01
w7a, 24 692	300	w7a.t, 25 057	1	0.01
w8a, 49 749	300	w8a.t, 14 951	1	0.01
S_s(Splice_scale), 1 000	60	S_s.t, 2 175	1	1.00
(I_s)Inonosphere_scale, 351	34	原训练集	1	0.10

在以下 3 组实验中，*Time* 表示每次训练所花费的总时间开销；*IRT* 表示求解最优超平面时所进行的迭代次数；*ST* 为预选后训练集的大小；*SVs* 表示支持向量的个数；*BSVs* 表示边界支持向量的个数；*Cor* 表示对测试集测试的准确率。

(1)只将标记为 2 的样本点作为训练集

标记为 2 的样本点为相邻边界点，一般分布在点集的相邻边界处，它们绝大部分为支持向量。各训练集的实验结果如表 2 所示。其中，*SVs* 与 *ST* 的比值接近或等于 1，由此说明所求的相邻边界点中绝大部分为支持向量，同时也表明运用相邻边界模型提取训练集的思路是正确的。但从实验结果可以看出，测试的准确率总体上不太高，原因在于通过相邻边界点求出的支持向量只是所有支持向量的一个子集，并不能代表原训练集。为了提高测试的准确率，有必要添加一些样本点以提高训练集的信息含量。

表 2　以标记为 2 的样本点作为训练集的实验结果

训练集	*Time*/s	*IRT*	*ST*	*SVs*	*BSVs*	*Cor*/(%)
ijcnn1	7.640	749	195	149	30	76.27
a1a	0.750	1 575	349	332	60	66.83
a9a	77.500	4 614	2 534	2 365	2 182	77.40
w7a	1.593	60	43	43	29	92.10
w8a	3.094	29	28	28	20	96.96
S_s	0.359	134	66	66	29	48.00
I_s	0.078	161	56	42	9	96.01

(2)将标记不为 0 的样本点作为训练集

标记为 1 的样本点大多分布在各自类别的边缘处，它们的加入能很好地概括原训练集的轮廓，增加训练集的信息含量，实验结果如表 3 所示。

表 3　以标记不为 0 的样本点作为训练集的实验结果

训练集	*Time*/s	*IRT*	*ST*	*SVs*	*BSVs*	*Cor*/(%)
ijcnn1	7.921	902	307	175	32	90.07
a1a	0.860	991	653	422	349	80.29
a9a	86.266	5 254	4 048	3 035	2 841	82.33
w7a	1.594	117	71	64	27	95.39
w8a	3.141	40	52	48	34	96.93
S_s	0.485	259	133	133	65	52.00
I_s	0.094	267	52	65	10	97.44

由表 2 和表 3 可以看出，标记为 1 的样本点的加入显著提高了测试的准确率，虽然训练时间和样本点的个数有所增加，但增加只是少量的，并不影响算法的整体性能。因此，本文提出的相邻边界模型预选算法将标记不为 0 的样本点作为最终的训练集。

(3)将所有的样本点作为训练集

为了证明该预选策略的有效性和高效性，表 4 记录了没有使用预选模块的 LIBSVM 的实验结果，其中，*total* 为训练数据集的大小。

表 4　原 LIBSVM 的实验结果

训练集	*Time*/s	*IRT*	*total*	*SVs*	*BSVs*	*Cor*/(%)
ijcnn1	200.000	16 713	49 996	3 154	2 440	98.99
a1a	1.125	1 458	1 605	640	558	84.38
a9a	387.000	27 674	32 561	11 465	10 911	85.06
w7a	75.340	2 204	24 692	1 490	824	98.66
w8a	123.400	3 553	49 749	2 846	2 490	98.16
S_s	2.485	1 650	1 000	998	437	52.00
I_s	0.094	398	351	81	15	98.86

由表 3 和表 4 可以看出，采用预选模块后 LIBSVM 仍然保持着较好的准确率，但相比于原 LIBSVM，准确率平均下降了 3.9%，主要原因是相邻边界点中支持向量只是所有支持向量的一部分，并不能完全代表整个训练集，表 4 中 *SVs* 是表 3 中 *SVs* 的 18 倍也能说明这一点。但在平均情况下，采样比为 177，即采样后满足条件 $n<<M$，说明改进的算法具有近似线性的时间复杂度，表 4 中训练时间 *Time* 是表 3 的 18.7 倍也能说明这一点，从采样比还能看出改进的算法训练时占用更少的内存。算法在时间和空间效率上的大幅度提高充分证明了本算法的有效性和高效性，也说明了该学习策略为解决大规模数据训练问题提供了一种有效的方法。

为了进一步验证本策略在解决大规模数据训练问题时的高效性和实用性，用改进的 LIBSVM 对 KDDCUP99 入侵检测数据进行了实验。

3.2　实验二

实验选用的入侵数据是 KDDCUP99 提供的所有数据的一个子集，在正式实验前需对 KDDCUP99 入侵检测数据进行一些处理：(1)对原入侵数据本身进行处理，主要是将字符型数值转化为数值型和对数值进行归一化处理；(2)将 KDDCUP99 数据文件格式转换成 LIBSVM 能识别的数据文件格式；(3)对训练集和测试集进行提取，根据实验目的将相同类型的数据归为一类，以便于进行二分类实验。处理后的实验用数据信息和入侵检测实验结果见表 5、表 6。表 6 中训练程序 Sample_svm 为改进的 LIBSVM，因为原 LIBSVM 对训练集进行训练时没有进行采样，所以采样后样本数属性记录着原训练集的大小。

表 5　训练数据和测试数据信息

攻击类型	训练集及其大小	评测类型	测试集及其大小
DOS	Normal+DOS，488 728 个	正确率 检测率	Normal+DOS，290 441 个 DOS，229 853 个
Probing	Normal+ Probing，101 378 个	正确率 检测率	Normal+Probing，64 754 个 Probing，4 166 个
R2L	Normal+ R2L，98 397 个	正确率 检测率	Normal+R2L，76 776 个 R2L，16 188 个
U2R	Normal+ U2R，97 323 个	正确率 检测率	Normal+U2R，60 816 个 U2R，228 个
所有攻击	Normal+所有，494 020 个	正确率 检测率	Normal+所有，311 029 个 所有攻击，250 435 个

表 6　入侵检测实验结果

攻击类型	训练程序	训练参数	训练时间/s	正确率/(%)	检测率/(%)	采样后样本数
DOS	Sample_svm	G=0.01 C=100	24.53	96.87	97.82	64
	LIBSVM		911.52	97.69	97.42	488 728
Probing	Sample_svm	G=0.01 C=50	4.38	95.85	88.05	21
	LIBSVM		82.88	98.47	79.59	101 378
R2L	Sample_svm	G=0.01 C=1 000	5.59	71.88	74.83	42
	LIBSVM		76.03	79.31	1.93	98 397
U2R	Sample_svm	G=0.01 C=1 000	6.33	95.11	89.91	90
	LIBSVM		22.16	99.65	8.77	97 323
所有攻击	Sample_svm	G=0.01 C=100	26.23	90.78	91.17	59
	LIBSVM		1 909.70	92.40	90.94	494 020

对表 6 的结果从以下 2 个方面进行分析比较：(1)通过与 LIBSVM 的比较来进一步验证本主动学习策略的高效性。可以看出，在相同的实验条件下，Sample_svm 的训练时间平均比 LIBSVM 快 29 倍，检测率也明显提高，特别是对 R2L 和 U2R 这 2 种攻击的检测。训练时间变快是因为采样后的训练集只占原训练集很小的一部分，表 6 中采样比最高为 8 373.2，最小为 1 081.4，平均为 4 852.2，满足条件 $n<<M$，算法具有线性的时间复杂度，因此，占用很小的内存空间。不足之处是正确率略有下降。(2)通过与其他同类实验结果进行比较来说明本算法具有一定的实用性。考虑到各种因素，比如实验的环境、条件和算法设计的复杂度，没有从严格的意义上与其他实验进行全方位的比较，这里只注重它们的检测率。1998 年 DARPA 入侵评测的结果中，对 DOS 攻击的检测率最好只有 65%，对 Probing 攻击的检测率最好为 97%，对 R2L 攻击的检测率最好小于 35%，对 U2R 攻击的检测率最好小于 70%。文献[5]列出了 KDDCUP99 获胜者的入侵评测结果，其中，对 DOS 的检测率为 97.1%，对 Probing 的检测率为 83.3%，对 R2L 的检测率为 8.4%，U2R 的检测率为 13.2%。由此可以看出，改进的 LIBSVM 算法性能及检测率都大大提高，尤其是对 R2L 和 U2R 这 2 种攻击。

4　结束语

本文基于相邻边界模型提出了一种高效的、适合大规模数据处理的支持向量预选算法，通过实验证明该主动学习策略可以大大缩短 LIBSVM 训练数据集的时间，减少内存消耗。如何进一步提高算法的正确率、分析何种分布的训练集采用本学习策略后采样比能满足条件 $n<<M$，是今后研究的重点。

(下转第 193 页)