ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.18, No.11, November 2007, pp.2719−2727
DOI: 10.1360/jos182719

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于向量集约简的精简支持向量机*

曾志强+, 高 济

(浙江大学 计算机科学与技术学院,浙江 杭州 310027)

Simplified Support Vector Machine Based on Reduced Vector Set Method

ZENG Zhi-Qiang+, GAO Ji

(College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China)

+ Corresponding author: E-mail: lbxzzq@163.com, http://www.zju.edu.cn

Zeng ZQ, Gao J. Simplified support vector machine based on reduced vector set method. *Journal of Software*, 2007,18(11):2719−2727. http://www.jos.org.cn/1000-9825/18/2719.htm

Abstract: Existing works of reduced support vector set method find the reduced set vectors based on solving an unconstrained optimization problem with multivariables, which may suffer from numerical instability or get trapped in a local minimum. In this paper, a reduced set method relying on kernel-based clustering is presented to simplify SVM (support vector machine) solution. The method firstly organizes support vectors in clusters in feature space, and then, it finds the pre-images of the cluster centroids in feature space to construct a reduced vector set. This approach is conceptually simpler, involves only linear algebra and overcomes the difficulties existing in the former reduced set methods. Experimental results on real data sets indicate that the proposed method is effective in simplifying SVM solution while preserving machine's generalization performance.

Key words: support vector machine; reduced vector set; kernel-based clustering; pre-image; optimal weight

摘 要: 目前的支持向量集约简法在寻找约简向量的过程中需要求解一个无约束的多参数优化问题,这样,像其他非线性优化问题一样,求解过程需要面对数值不稳定或局部最小值问题.为此,提出了一种基于核聚类的SVM(support vector machine)简化方法.此方法首先在特征空间中对支持向量进行聚类,然后寻找特征空间中的聚类中心在输入空间中的原像以形成约简向量集.该方法概念简单,在简化过程中只需求解线性代数问题,从而解决了现存方法存在的瓶颈问题.实验结果表明,该简化法能够在基本保持 SVM 泛化性能的情况下极大地约简支持向量,从而提高 SVM 的分类速度.

关键词: 支持向量机;约简向量集;核聚类;原像;最佳权值

中图法分类号: TP18 **文献标识码**: A

支持向量机(support vector machine,简称SVM)[1]是在统计学习理论基础上发展起来的一种新的机器学习方法,它基于结构风险最小化原则,在解决小样本、非线性及高维模式识别问题中表现出许多特有的优势,并能推广到函数拟和等其他机器学习问题中.近年来,SVM在手写数字识别、语音识别、文本分类[2]等许多实际应用中都取得了成功.

* Supported by the National Basic Research Program of China under Grant No.2003CB317000 (国家重点基础研究发展计划(973))
Received 2006-10-17; Accepted 2006-11-14

然而,SVM存在着一个明显的缺点,即它的分类速度取决于支持向量的数目,如果支持向量数目很大,则SVM的分类速度很慢[3],这在很大程度上限制了SVM的应用.为解决此问题,许多研究者通过寻找一个包含较少支持向量的约简向量集代替原有的支持向量集来提高分类速度.文献[3]削减那些在特征空间中能被其他支持向量线性表示的冗余支持向量,削减完毕后剩余的支持向量即所求的约简向量,然后修改约简向量所对应的权值,使SVM判定函数保持不变.这种方法能够在保持分类精度不变的情况下减少支持向量的数目,从而达到简化SVM、提高分类速度的目的.然而实验结果表明,采用此种方法,支持向量的约简率不高.为了提高支持向量约简率,Scholkopf等人[4,5]采用迭代构建新的向量,作为约简向量的方法来简化SVM.虽然此类简化法导致一定的分类精度受损,但却取得了较大的支持向量约简率,极大地加快了SVM的分类速度.然而,此类方法求解过程中存在着严重的瓶颈:寻找约简向量过程中需要解决一个多参数的无约束优化问题,这样,像其他非线性优化问题一样,求解过程需要面对数值不稳定或局部最小值问题[6].另一方面,此类简化法最终所获得的约简向量缺乏明显的物理意义.为此,本文提出了一种新的SVM简化法.这种简化方法概念简单,约简后的向量具有明确的物理意义,在简化过程中只需求解线性代数问题,从而解决了局部最小值问题.实验结果表明,此简化法在基本保持向量机泛化性能的同时极大地约简了支持向量.

1 SVM 及其简化方法

1.1 支持向量机

训练SVM的本质就是求解一个最优分类超平面问题.给定训练样本(x_i,y_i),$i=1,\ldots,l$,其中,$x_i\in R^h$,$y_i\in\{1,-1\}$,求解最优分类超平面可以转化为优化一个二次规划问题[1]:

$$\begin{cases}\min \dfrac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}y_iy_j\alpha_i\alpha_jk(x_i,x_j)-\sum_{i=1}^{l}\alpha_i\\ \text{s.t. }\sum_{i=1}^{l}y_i\alpha_i=0,\ \forall i:0\le\alpha_i\le C\end{cases}\tag{1}$$

其中,α_i为训练样本x_i所对应的拉格朗日乘子(权值),参数C为惩罚因子,$k(x_i,x_j)=\varphi(x_i)^T\varphi(x_j)$为核函数,它对应于采用非线性映射 $\varphi:R^h\mapsto F$ 将训练样本从输入空间映射到某一特征空间 F,在该特征空间中,样本是线性可分的.对于两类分类问题,SVM 的判定函数形式如下:

$$f(x)=\mathrm{sgn}\left(\sum_{i=1}^{N_S}\alpha_ik(\boldsymbol{x}_i,\boldsymbol{x})+b\right)\tag{2}$$

其中$\boldsymbol{x}_i$,$i=1,\ldots,N_S$就是所谓的支持向量,它们对应的拉格朗日乘子α_i不等于 0.$\boldsymbol{x}$为待分类的向量,N_s为支持向量的数量,b为偏置.从式(2)可以看出,判定一个未知类别的样本所需要的时间与支持向量的数目成正比.因此,削减支持向量的数量能够有效地提高向量机的分类速度.

1.2 支持向量机简化法

SVM 训练所得的分类超平面所对应的向量$\boldsymbol{\psi}$在形式上表示为所有支持向量在特征空间中的线性组合,

$$\boldsymbol{\psi}=\sum_{i=1}^{N_S}\alpha_i\varphi(\boldsymbol{x}_i)\tag{3}$$

SVM 简化法试图采用一个约简的向量集来代替所有的支持向量,

$$\boldsymbol{\psi}'=\sum_{i=1}^{N_Z}\beta_i\varphi(\boldsymbol{z}_i)\tag{4}$$

其中,$\{z_1,\ldots,z_{N_Z}\}\in R^h$ 就是约简向量集,$\beta_i\in R$为约简向量$\boldsymbol{z}_i$所对应的权值,N_Z为约简向量集所包含的向量个数,并且$N_Z<N_S$.这样,可用$\boldsymbol{\psi}'$代替$\boldsymbol{\psi}$来判定未知类别的向量$\boldsymbol{x}$,此时,SVM的判定函数形式如下:

$$f(x)=\mathrm{sgn}\left(\sum_{i=1}^{N_Z}\beta_ik(\boldsymbol{z}_i,\boldsymbol{x})+b\right)\tag{5}$$

SVM简化法的目标就是在尽量减小分类精度损失的前提下,寻找最小的$N_Z<<N_S$和对应的约简向量集,形成一个精简的SVM来提高分类速度[4].

Downs[3]通过删除那些可被其他支持向量线性表达的冗余支持向量达到简化SVM的目的.虽然这种方法

保持了原向量机的分类精度,但是实验结果表明,它对支持向量的削减率不高.Scholkopf和Smola等人[4,5]提出了另一类SVM简化方法,此类方法首先构造一个新的约简向量及其对应权值$(\boldsymbol{z}_1,\beta_1)$来近似式(3)中的向量$\boldsymbol{\psi}$,接着,迭代地构建$(\boldsymbol{z}_{m+1},\beta_{m+1})$来近似向量$\boldsymbol{\psi}_m$.$\boldsymbol{\psi}_m$的形式如下:

$$\boldsymbol{\psi}_m=\sum_{i=1}^{N_S}\alpha_i\varphi(\boldsymbol{x}_i)-\sum_{i=1}^{m}\beta_i\varphi(\boldsymbol{z}_i) \tag{6}$$

由于不可能精确地找到向量$\boldsymbol{z}_m$和对应权值β_m使向量$\boldsymbol{\psi}_m$为 0,所以只能通过非线性优化来寻找最小的δ.δ的形式如下式所示:

$$\delta=\|\boldsymbol{\psi}_{m-1}-\beta_m\varphi(\boldsymbol{z}_m)\|^2 \tag{7}$$

对于某些特殊的核函数,例如高斯核函数$k(x,y)=\exp(-\|x-y\|^2/2\sigma^2)$,Scholkopf等人采用不动点迭代法来寻找约简向量$\boldsymbol{z}$,设式(7)的导数为 0,求约简向量$\boldsymbol{z}$的迭代公式如下[4,5]:

$$\boldsymbol{z}_{n+1}=\frac{\sum_{i=1}^{N_S}\alpha_i\exp(-\|\boldsymbol{x}_i-\boldsymbol{z}_n\|^2/(2\sigma^2))\boldsymbol{x}_i}{\sum_{i=1}^{N_S}\alpha_i\exp(-\|\boldsymbol{x}_i-\boldsymbol{z}_n\|^2/(2\sigma^2))} \tag{8}$$

然而,正如文献[6]中所提到的,这个迭代过程容易陷入数值不稳定或局部最小值问题.因此,求解每一个约简向量都必须采用不同的初始值重复迭代过程,以避免陷入局部最小值.

2 基于核聚类的支持向量机简化法

本文所提出的 SVM 简化法的基本思想就是在特征空间中对正、负两类支持向量分别进行聚类,然后用聚类后所形成的簇的质心代替簇内的支持向量来简化 SVM.这样,特征空间中所有簇的质心组成的集合就形成了约简向量集,采用约简向量的线性组合来近似式(3)中的$\boldsymbol{\psi}$.此时,约简后的分类超平面$\boldsymbol{\psi}'=\sum_{i=1}^{N_Z}\beta_i\varphi(\boldsymbol{z}_i)$,其中,$\varphi(\boldsymbol{z}_i)$,$i=1,\ldots,N_Z$为特征空间中所有簇的质心,$\beta_i$为对应的权值,$N_Z$为簇的数量,即约简向量的数量.从机械系统的角度来看,在特征空间中,每个支持向量都对分类超平面施加了影响,它们的综合影响使分类超平面保持目前的平衡状态,如果几个支持向量对分类超平面的影响被一个约简向量等价地替代,则系统的平衡状态不会改变[7].本简化法就是试图将簇内支持向量对分类超平面所施加的综合影响等价地用簇的质心的影响来替代,从而在精简支持向量的同时仍然保持超平面的平衡状态,维持向量机的泛化性能.

2.1 核聚类算法

文献[8]提出了一种简单的无监督聚类(unsupervised clustering,简称UC)算法.与传统的k-means算法预先指定类别数不同,UC 算法根据预先给定的聚类半径来对数据进行聚类,它具有较高的聚类速度,因此,本文采用UC算法来对支持向量进行聚类.然而,UC算法工作在输入空间,而SVM简化法操作在特征空间,显然,在输入空间中通过 UC 算法所获得的簇质心不适宜作为特征空间中的约简向量.因此,本文对 UC 算法进行了扩展,使其能够对特征空间中的数据进行聚类,扩展后的算法称为核聚类算法(kernel-based clustering,简称 KUC). KUC 算法描述如下:

假设待聚类的正(负)支持向量集为$\boldsymbol{X}=\{x_1,x_2,\ldots,x_m\}$,$\boldsymbol{x}_i\in R^h$,$i=1,\ldots,m$,聚类半径设为$r$,$\varphi$为非线性映射,它将输入空间中的点映射到特征空间$F$.

1) $C_1=\{\varphi(x_1)\}$,$O_1=\varphi(x_1)$,$Cluster\_num=1$,$\boldsymbol{Z}=\{x_2,\ldots,x_m\}$.

2) 如果 $\boldsymbol{Z}=\varnothing$,则 STOP.

3) 选择样本$\boldsymbol{x}_i\in\boldsymbol{Z}$,从已有的质心中寻找与$\varphi(\boldsymbol{x}_i)$距离最近的质心$O_j$,即

$$O_j=\arg\min_{j}^{Cluster\_num}{}_{k=1}\ d(\varphi(\boldsymbol{x}_i),O_k) \tag{9}$$

4) 如果$d(\varphi(\boldsymbol{x}_i),O_j)\le r$,则将$\varphi(\boldsymbol{x}_i)$加入类$C_j$,即$C_j=C_j\cup\{\varphi(\boldsymbol{x}_i)\}$,类$C_j$的质心调整为

$$O_j=\frac{n_j\times O_j+\varphi(\boldsymbol{x}_i)}{n_j+1} \tag{10}$$

其中,n_j为类C_j所包含的样本数目.调整$n_j=n_j+1$, go to Step 6).

5) 如果$d(\varphi(\boldsymbol{x}_i),O_j)>r$,增加一个新类:

$$Cluster\_num=Cluster\_num+1, C_{Cluster\_num}=\{\varphi(\boldsymbol{x}_i)\}, O_{Cluster\_num}=\{\varphi(\boldsymbol{x}_i)\}.$$

6) $\boldsymbol{Z}=\boldsymbol{Z}-\{\boldsymbol{x}_i\}$, go to Step 2).

在第 3)步中,如下计算$\varphi(\boldsymbol{x}_i)$和第$k$个类的质心$O_k=(1/n_k)\sum_{p=1}^{n_k}\varphi(\boldsymbol{x}_{k_p})$之间的距离:

$$d(\varphi(\boldsymbol{x}_i),O_k)=\sqrt{k(\boldsymbol{x}_i,\boldsymbol{x}_i)-\frac{2}{n_k}\sum_{p=1}^{n_k}k(\boldsymbol{x}_i,\boldsymbol{x}_{k_p})+\frac{1}{n_k^2}\sum_{p,q=1}^{n_k}k(\boldsymbol{x}_{k_p},\boldsymbol{x}_{k_q})} \tag{11}$$

其中,n_k表示$|C_k|$, $\varphi(\boldsymbol{x}_{k_i}), i=1,...,n_k$ 为特征空间中属于类C_k的支持向量.

从式(10)可以看出,通过 KUC 算法所获得的类质心为特征空间中从属于本类的所有支持向量的均值.这是假定所有的支持向量都对质心施加了相同的影响,没有考虑支持向量对应的权值.为解决此问题,调整质心的表达式如下:

$$O_k=\sum_{i=1}^{n_k}b_{k_i}\varphi(\boldsymbol{x}_{k_i}),\ b_{k_i}=\alpha_{k_i}/\sum_{i=1}^{n_k}\alpha_{k_i},\ k=1,\ldots,Cluster\_num \tag{12}$$

其中, $\alpha_{k_i}, i=1,...,n_k$ 为特征空间中从属于类C_k的支持向量所对应的权值.

2.2 寻找质心的原像

从式(12)可以看出,特征空间中的簇质心在形式上表示为簇内支持向量的线性组合,由于映射φ未知,直接将此表达式代入SVM判定函数并不能达到简化SVM的目的,因此,本文试图寻找簇C_k,$k=1,\ldots,Cluster_num$的质心O_k在输入空间的原像z_k,使得$\varphi(z_k)=O_k$.然而,由于非线性映射$\varphi^{-1}:F\to R^h$是未知的,所以不可能精确地得到质心在输入空间中的原像$z_k=\varphi^{-1}(O_k)$,只能通过其他方法得到它的近似解.本文采用文献[9]中的策略,利用输入空间和特征空间之间的距离关系来寻找质心O_k在输入空间的原像z_k.

要确定原像z_k,首先必须建立输入空间和特征空间之间的距离关系,虽然目前只能对等方性核函数$k(x,y)=K(\|x-y\|)$(例如高斯核函数)确立这种距离关系,但考虑到此类核函数在实际应用中使用最为广泛,因此,本简化法仍具有显著的实用性.

在特征空间中,如下计算任意一样本点x_i到质心O_k的距离:

$$\tilde{d}_i^2(O_k,\varphi(\boldsymbol{x}_i))=k(\boldsymbol{x}_i,\boldsymbol{x}_i)-2\sum_{p=1}^{n_k}b_{k_p}k(\boldsymbol{x}_{k_p},\boldsymbol{x}_i)+\sum_{p,q=1}^{n_k}b_{k_p}b_{k_q}k(\boldsymbol{x}_{k_p},\boldsymbol{x}_{k_q}) \tag{13}$$

对于高斯核函数而言,$\boldsymbol{x}_i$与O_k的原像z_k在特征空间中的距离$\tilde{d}_i^2(\varphi(z_k),\varphi(\boldsymbol{x}_i))$与输入空间中的距离$d_i^2(z_k,\boldsymbol{x}_i)$维持如下关系[10]:

$$\begin{aligned}\tilde{d}_i^2(\varphi(z_k),\varphi(\boldsymbol{x}_i))&=\|\varphi(z_k)-\varphi(\boldsymbol{x}_i)\|^2=k(z_k,z_k)-2k(z_k,\boldsymbol{x}_i)+k(\boldsymbol{x}_i,\boldsymbol{x}_i)\\&=2-2\exp(-\|z_k-\boldsymbol{x}_i\|^2/(2\sigma^2))=2-2\exp(-d_i^2(z_k,\boldsymbol{x}_i)/(2\sigma^2))\\&\Rightarrow d_i^2(z_k,\boldsymbol{x}_i)=-2\sigma^2\ln\left(1-\frac{1}{2}\tilde{d}_i^2(\varphi(z_k),\varphi(\boldsymbol{x}_i))\right)\end{aligned} \tag{14}$$

因为$\tilde{d}_i^2(O_k,\varphi(\boldsymbol{x}_i))=\tilde{d}_i^2(\varphi(z_k),\varphi(\boldsymbol{x}_i))$,所以根据式(13)、式(14)可以得到$d_i^2(z_k,\boldsymbol{x}_i)$,即样本点$\boldsymbol{x}_i$与$O_k$的原像$z_k$在输入空间中的距离.通常,样本点与其近邻的距离在确定样本点位置的过程中起着至关重要的作用,所以在求z_k的过程中,我们主要考虑质心O_k与其特征空间中的n_k个近邻$\{\varphi(x_{k_1}),\varphi(x_{k_2}),...,\varphi(x_{k_{n_k}})\}$在输入空间中的距离(用二次方来衡量),其中,$\{\varphi(x_{k_1}),\varphi(x_{k_2}),...,\varphi(x_{k_{n_k}})\}$就是从属于类$C_k$的支持向量在特征空间中的像所组成的集合.定义向量

$$d^2=[d_1^2,d_2^2,...,d_{n_k}^2]^T \tag{15}$$

其中,$d_i, i=1,\ldots,n_k$为O_k的原像z_k和它的近邻$\boldsymbol{x}_{k_i}$在输入空间中的距离.文献[9,11]采用某个未知坐标的点和其他点之间的距离约束来确定此点在空间中的坐标,借鉴其思想来寻找O_k在输入空间中的原像z_k.对于O_k在特征空间中的n_k个近邻$\{\varphi(x_{k_1}),\varphi(x_{k_2}),...,\varphi(x_{k_{n_k}})\}$,确定此$n_k$个近邻在输入空间中的原像$\{x_{k_1},x_{k_2},...,x_{k_{n_k}}\}\in R^h$的均值$\bar{x}=(1/n_k)\sum_{i=1}^{n_k}\boldsymbol{x}_{k_i}$,并构建一个新的坐标系.首先创建一个$h\times n_k$的矩阵$X=[x_{k_1},x_{k_2},...,x_{k_{n_k}}]$和一个$n_k\times n_k$的中心

矩阵:

$$H = I - \frac{1}{n_k}11^T \tag{16}$$

其中,I是一个$n_k \times n_k$的单位矩阵,$1=[1,1,\ldots,1]^T$为$n_k \times 1$的向量,则矩阵XH是以$\bar{x}$为中心的$h \times n_k$中心矩阵:

$$XH = [x_{k_1} - \bar{x}, x_{k_2} - \bar{x}, ..., x_{k_{n_k}} - \bar{x}] \tag{17}$$

假设矩阵XH的秩为q,对其进行奇异值分解:

$$XH = [E_1, E_2]\begin{bmatrix} \Lambda_1 & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} = E_1 \Lambda_1 V_1^T = E_1 \Gamma \tag{18}$$

其中,$E_1=[e_1,e_2,\ldots,e_q]$为一组标准正交列向量$\boldsymbol{e}_i$组成的$h \times q$矩阵,$\Gamma = \Lambda_1 V_1^T = [c_1, c_2, ..., c_{n_k}]$为一个$q \times n_k$矩阵,列向量$\boldsymbol{c}_i$为向量$\boldsymbol{x}_{k_i} - \bar{x}$在$E_1$上的投影,此时,$\| c_i \|^2 = \| \boldsymbol{x}_{k_i} - \bar{x} \|^2, i = 1,..., n_k$,定义一个$n_k \times 1$的向量$\boldsymbol{d}_0^2 = [\| c_1 \|^2, \| c_2 \|^2, ..., \| c_{n_k} \|^2]^T$.显然,为了获得较为精确的原像$z_k$,距离$d^2(z_k, \boldsymbol{x}_{k_i}), i = 1,..., n_k$应尽可能地等于式(15)中的值,即

$$d^2(z_k, \boldsymbol{x}_{k_i}) \approx d_i^2,\ i = 1,..., n_k \tag{19}$$

定义$\tilde{c} \in R^{q \times 1}$并且$E_1\tilde{c} = z_k - \bar{x}$,则

$$d_i^2 \approx \| z_k - \boldsymbol{x}_{k_i} \|^2 = \| (z_k - \bar{x}) - (\boldsymbol{x}_{k_i} - \bar{x}) \|^2 = \| \tilde{c} \|^2 + \| c_i \|^2 - 2(z_k - \bar{x})^T (\boldsymbol{x}_{k_i} - \bar{x}),\ i = 1,..., n_k \tag{20}$$

采用与文献[11]类似的步骤,首先对等式(20)从 1 累加到n_k,由于XH为中心矩阵,所以式(20)中内积项的累加和为 0.累加完的等式如下式所示:

$$\sum_{i=1}^{n_k} d_i^2 = n_k \| \tilde{c} \|^2 + \sum_{i=1}^{n_k} \| c_i \|^2 \Rightarrow \| \tilde{c} \|^2 = \frac{1}{n_k}\sum_{i=1}^{n_k} (d_i^2 - \| c_i \|^2),\ i = 1,..., n_k \tag{21}$$

将(21)式中$\| \tilde{c} \|^2$的表达式代入式(20)并重新排列可得,

$$2(\boldsymbol{x}_{k_i} - \bar{x})^T (z_k - \bar{x}) = \| c_i \|^2 - d_i^2 - \frac{1}{n_k}\sum_{i=1}^{n_k} (\| c_i \|^2 - d_i^2),\ i = 1,..., n_k \tag{22}$$

采用矩阵的形式来表达式(22)可得,

$$2\Gamma^T \tilde{c} = (\boldsymbol{d}_0^2 - d^2) - \frac{1}{n_k}11^T (\boldsymbol{d}_0^2 - d^2) \tag{23}$$

由于Γ为中心矩阵,所以$\Gamma 11^T = 0$.对式(23)进行适当变换可得,

$$\tilde{c} = \frac{1}{2}(\Gamma\Gamma^T)^{-1}\Gamma(\boldsymbol{d}_0^2 - d^2) = \frac{1}{2}\Lambda_1^{-1} V_1^T (\boldsymbol{d}_0^2 - d^2) \tag{24}$$

最后,将$\tilde{c}$转换回输入空间中的原始坐标系,可以得到质心O_k在输入空间中的原像的近似值:

$$z_k = \frac{1}{2}E_1 \Lambda_1^{-1} V_1^T (\boldsymbol{d}_0^2 - d^2) + \bar{x} \tag{25}$$

2.3 确定约简向量权值

获得聚类质心在输入空间中的原像后,接下来的目标就是寻找约简向量的最佳权值β_k,使$\sum_{i=1}^{n_k} \alpha_{k_i} \varphi(\boldsymbol{x}_{k_i})$和它的近似值$\beta_k \varphi(z_k)$尽量相等.定义

$$d(\beta_k) = \| \beta_k \varphi(z_k) - \sum_{i=1}^{n_k} \alpha_{k_i} \varphi(\boldsymbol{x}_{k_i}) \|^2,\ k = 1,..., Cluster\_num \tag{26}$$

其中,z_k为类C_k的质心的原像,$\boldsymbol{x}_{k_i}$, $i=1,\ldots,n_k$为从属于类C_k的支持向量,α_{k_i}为对应的权值.对式(26)关于β_k求导,以获得使$d(\beta_k)$取最小值的最佳权值β_k,即令$\nabla_{\beta_k}(d(\beta_k)) = 0$,可得

$$\beta_k = \sum_{i=1}^{n_k} \alpha_{k_i} k(z_k, \boldsymbol{x}_{k_i}) / k(z_k, z_k),\ k = 1,..., Cluster\_num \tag{27}$$

对于高斯核函数而言,$k(z_k,z_k)=1$,此时,$\beta_k = \sum_{i=1}^{n_k} \alpha_{k_i} k(z_k, \boldsymbol{x}_{k_i})$.

2.4 衡量分类超平面的变化

通常情况下,简化后所得的分类超平面与原始分类超平面之间存在着差异,这种差异可能导致精简 SVM 泛

化性能的下降,差异越大,泛化性能下降得越多.为了使简化后的 SVM 保持一定的分类精度,有必要对简化过程中所造成的分类超平面的变化进行监控.我们定义下式来衡量简化前后两个分类超平面之间的差异:

$$\frac{\|\boldsymbol{\psi}-\boldsymbol{\psi}'\|^2}{\|\boldsymbol{\psi}\|^2}=\frac{\|\sum_{i=1}^{N_S}\alpha_i\varphi(\boldsymbol{x}_i)-\sum_{p=1}^{N_Z}\beta_p\varphi(\boldsymbol{z}_p)\|^2}{\|\sum_{i=1}^{N_S}\alpha_i\varphi(\boldsymbol{x}_i)\|^2}=1+\frac{\sum_{p,q=1}^{N_Z}\beta_p\beta_q k(\boldsymbol{z}_p,\boldsymbol{z}_q)-2\sum_{i=1}^{N_S}\sum_{p=1}^{N_Z}\alpha_i\beta_p k(\boldsymbol{x}_i,\boldsymbol{z}_p)}{\sum_{i,j=1}^{N_S}\alpha_i\alpha_j k(\boldsymbol{x}_i,\boldsymbol{x}_j)} \tag{28}$$

2.5 时间复杂度分析

由式(2)可知,SVM判断一个未知类别的样本所需要的时间和支持向量的数目成正比.假设精简前后SVM的支持向量分别为N_S,N_Z,以核函数的计算次数来度量,则原始SVM预测一个样本的时间复杂度为$O(N_S)$,精简SVM对应的时间复杂度为$O(N_Z)$,由于$N_Z<<N_S$,所以精简SVM比原始SVM具有更快的分类速度.

2.6 算法描述

本算法的基本思路是,在设定差异阈值对简化前后分类超平面变化的最大值作出限制的情况下(即限制泛化性能损失),迭代地增加聚类半径的大小来尽可能地约简支持向量,在每次迭代中,计算所形成的超平面和原始分类超平面之间的差异,如果二者之间的差异超过差异阈值τ,则迭代简化过程终止,取上一次迭代所得的简化 SVM 作为最终的精简 SVM,否则,增大聚类半径以进一步约简 SVM.算法总体框架描述如下:

输入:正类支持向量集SV^+,负类支持向量集SV^-,差异阈值τ.

输出:精简支持向量机 *FinalSVM*.

函数:

getClusterCenters(*S*,*r*):返回用 KUC 算法对数据集 *S* 进行聚类后所获得的类质心集合(聚类半径为 *r*).

getPreimage(***c***):返回特征空间中的向量 ***c*** 在输入空间中的原像.

getOptimalWeight(***rsv***):返回约简向量 ***rsv*** 对应的最佳权值(根据式(27)计算).

getReducedSVM(***RSV***,β):返回约简向量集 ***RSV*** 和对应权值集合β所形成的精简 SVM.

getDifference(SVM_1,SVM_2):返回向量机SVM_1和SVM_2所代表的超平面之间的差异值(根据式(28)计算).

算法:

1. Initialize r,λ,*FinalSVM*; //初始化聚类半径 r、步长λ,精简支持向量机 *FinalSVM*
2. C^+:=*getClusterCenters*(SV^+,r);C^-:=*getClusterCenters*(SV^-,r);
3. C:=$C^+\cup C^-$;
4. For i:=1 to $|C|$
5. {rsv_i:=*getPreimage*(c_i); //c_i代表集合C中的元素
6. β_i:=*getOptimalWeight*(rsv_i);
7. β:=$\beta\cup\{\beta_i\}$;*RSV*:=*RSV*$\cup\{rsv_i\}$;}
8. *RSVM*:=*getReducedSVM*(*RSV*,β);
9. δ:=*getDifference*(*RSVM*,*OriginalSVM*); //*OriginalSVM* 代表原始 SVM
10. If ($\delta>\tau$) then
11. go to Step 15
12. Else
13. {*FinalSVM*:=*RSVM*;
14. β:=$\varnothing$,*RSV*:=$\varnothing$,r:=$r+\lambda$; go to Step 2}
15. Return *FinalSVM*

2.6.1 初始化

显然,聚类半径越大,所形成的簇的数量就越少,支持向量的削减率也就越高,精简前后分类超平面之间差异就越大,泛化性能损失也就越多.因此,聚类半径的大小在向量机的约简率和泛化性能损失之间起着折衷的作用.为了避免初始聚类半径过大导致初次迭代即造成SVM泛化性能下降过多而使算法立即满足停机条件而终止,采用以下方法来初始化一个较小的聚类半径.首先从正类支持向量中随机地选取一个较小子集

$\boldsymbol{\omega}=\{x_1,x_2,\ldots,x_l\}$,然后计算$\boldsymbol{\omega}$中的向量在特征空间$F$中的平均距离,计算公式如下:

$$d^+=\frac{1}{l(l-1)}\sum_{i=1}^{l}\sum_{j=1,j\neq i}^{l}d(\varphi(\boldsymbol{x}_i),\varphi(\boldsymbol{x}_j))=\frac{1}{l(l-1)}\sum_{i=1}^{l}\sum_{j=1,j\neq i}^{l}\sqrt{k(\boldsymbol{x}_i,\boldsymbol{x}_i)-2k(\boldsymbol{x}_i,\boldsymbol{x}_j)+k(\boldsymbol{x}_j,\boldsymbol{x}_j)} \tag{29}$$

d^+用来近似所有正类支持向量在F中的平均距离.用同样的方法可以获得所有负类支持向量在F中的平均距离d^-,然后将聚类半径的初始值设为 0.25×min(d^+,d^-).通常情况下,步长λ可以设定为聚类半径初始值的十分之一、差异阈值τ不超过 0.5,以保持精简SVM的泛化性能,精简支持向量机FinalSVM的初始值设为原始支持向量机.

2.6.2 小类别问题

对于一个类质心而言,它的近邻就是在特征空间中从属于这个类别的支持向量.由第 2.2 节可知,寻找类质心在输入空间中的原像主要依赖于质心及其近邻之间的距离约束.因此,如果从属于某个类的支持向量太少,则求解该类质心原像的过程中就会因为缺乏足够的近邻距离约束而导致最终所获得的质心原像准确度不高,从而影响了精简 SVM 的泛化性能.为解决此问题,对于包含元素较少的类别(不超过 4 个),我们不求解其质心的原像,而是将此类包含的所有支持向量直接作为约简向量参与形成精简 SVM.

3 实验结果及分析

我们用VC++ 6.0 和Matlab 7.0 实现了所提出的SVM简化算法.LIBSVM(a library for support vector machines) 2.71[12]被选作为标准的SVM训练算法.1999 数据挖掘竞赛所使用的入侵检测数据集[13]和其他 5 个数据集[14]被选择作为实验数据集.在所有实验中,LIBSVM的参数C和高斯核函数的参数g的选取是从实验数据集一随机抽取的较小子集上采用 10 次交叉测试所得结果的最优值.实验所用机器为PC机(P4 3.5GHZ,1G RAM),操作系统为Windows 2003 Server.

3.1 实验1

本实验所采用的入侵检测数据集是一批网络连接记录集,大约有 500 万条连接记录,其中含有大量的正常网络流量和各种攻击,具有很强的代表性.由于原始数据集过于庞大,所以只有两个具有代表性的数据集被选取作为实验数据集:一个名为 10Percent(训练集),包含 494 020 条记录;另一个名为 Correct(测试集),包含 311 029 条记录.本数据集包含 7 个符号属性,而 SVM 只能处理数值属性,因此,必须将符号属性转换为数值属性,我们采用文献[15]中的方法进行转换,处理完符号属性后将所有的属性值都规格化到[0,1]区间.

入侵检测训练数据集包含正常网络流量数据和 22 个攻击类别,测试集除了正常流量数据以外,还包含 38 个攻击类型.本实验将这两个数据集按照大的类型划分成 5 类,形成新的训练集和测试集,见表 1.我们采用 1-vs-rest 方式来训练这个多类分类问题,获得对应类别的 SVM 再对其进行简化.

Table 1 Statistics of training and test data

表 1 训练及测试数据的统计数字

Class Name	ClassTag	# Training	# Test
Normal	1	97 277	60 593
Probe	2	4 107	4 166
DOS	3	391 458	229 851
U2R	4	52	230
R2L	5	1 126	16 189

实验结果见表 2.表 2 中第 1 列为不同的差异阈值τ,第 3 列~第 7 列为对应 2 类问题的精简SVM的支持向量数、在测试集上的错误率和测试时间,第 8 列为 5 类问题的精简SVM的支持向量数、在测试集上的错误率和测试时间,括号内的数字为对应的支持向量削减率(约简率).最后一列展示在相同支持向量削减率的情况下,采用Scholkopf的方法[4,5]所获得的精简SVM在测试集上的错误率及测试时间(5 类问题).Scholkopf方法的最终结果是多次采用不同初始值重复简化过程所获得的结果中的最优值.第 2 行~第 4 行分别对应原始SVM的支持向量数、测试错误率及在测试集上的运行时间.

Table 2 Reduction rate, generalization performance and test speed of simplified SVM under different τ

表 2 不同τ值下的精简支持向量机约简率、泛化性能及测试速度

τ/Class		1	2	3	4	5	5-class	Scholkopf's method
Original SVM	# SVs	5 813	344	3 479	97	1 058	10 791 (0)	
	Errors rate (%)	7.84	0.61	2.36	0.07	5.22	3.22	
	Test time (s)	559	112	365	20	118	1174	
0.1	# RSVs	2 231	175	1 468	29	401	4 304 (60%)	4 304 (60%)
	Errors rate (%)	7.73	0.61	2.36	0.08	5.2	3.2	3.44
	Test time (s)	287	35	221	13	60	616	616
0.3	# RSVs	213	79	167	10	221	690 (94%)	690 (94%)
	Errors rate (%)	8.03	0.62	2.36	0.08	5.22	3.26	8.8
	Test time (s)	44	24	33	11	39	151	151
0.45	# RSVs	150	27	94	5	52	328 (97%)	328 (97%)
	Errors rate (%)	8.47	1.31	2.42	1.13	5.22	3.71	24.72
	Test time (s)	31	15	25	11	18	100	100

从表 2 可以看出,随着差异阈值取值的增大,本简化法所获得的精简 SVM 约简率升高,它们在测试集上的分类速度也随之提高.虽然精简 SVM 的泛化性能随着差异阈值的增大而有所降低,但与约简率相比,泛化性能的损失微乎其微.当差异阈值取 0.45 时,本简化法在对支持向量取得高达 97%削减率的同时,却仅有 0.49%的分类精度损失.与此同时,精简 SVM 在测试集上的分类速度却是原 SVM 的 11.7 倍.这说明对于入侵检测数据集,本文所提出的简化方法在极大削减支持向量的同时,基本上保持了原 SVM 的分类精度,极大地提高了 SVM 的分类效率,解决了 SVM 应用于入侵检测系统所存在的速度瓶颈问题.

与 Scholkopf 简化法相比,在 60%约简率的情况下,本精简 SVM 的分类精度甚至高于原始 SVM,而 Scholkopf 简化法在相同约简率下却造成了 0.22%的分类精度损失;在 94%和 97%的约简率下,本精简 SVM 的分类精度损失分别为 0.04%和 0.49%,而 Scholkopf 简化法所获得的精简 SVM 对应的分类精度损失却高达 5.58%和 21.5%.这说明在相同约简率下,本简化法所获得的精简 SVM 具有更好的泛化性能.由于精简 SVM 在相同约简率下具有相同的约简向量数量,因此,两种方法所获得的具有相同约简率的精简 SVM 在测试集上的分类时间相同.

另一方面,Scholkopf 简化法在求解过程中需要预先确定约简率,在约简率确定的情况下,寻找每个约简向量仍要采用不同的初始值重复式(8)的迭代过程多次,以避免陷入局部最小,代价高昂并且不一定能找到最优值.而本简化法不需要确定约简率,只需预先确定差异阈值,在寻找约简向量过程中只需求解线性代数问题,因此不存在局部最小值问题,一旦差异阈值确定,所获得的结果就是唯一的.

最后,Scholkopf 简化法所获得的约简向量缺乏明确的物理意义.而本简化法所获得的约简向量可以看作属于同一类别的距离较近的几个支持向量在特征空间中的代表,具有明确的物理意义.

3.2 实验2

我们进一步在其他 5 个数据集[14]上做了实验.这 5 个数据集分别是A1a,Mushroom(随机选择 5 000 条记录作为训练集,其余作为测试集),W1a,Letter(字母"N"为一类,其余字母为另一类),DNA(类型 2 为一类,其余为另一类).实验结果见表 3.

Table 3 Reduction rate and generalization performance of simplified SVM on five data sets

表 3 精简支持向量机在 5 个数据集上的约简率及泛化性能

Data set	Original SVM		Simplified SVM (our method)			Scholkopf's method
	# SVs	Error rate (%)	τ	# RSVs	Error rate (%)	Error rate (%)
A1a	630	15.6	0.45	41	15.9	17.1
Mushroom	130	0	0.15	23	0	0
W1a	149	2.6	0.25	8	2.6	3.8
Letter	744	0.9	0.1	97	1	5.4
DNA	516	5.5	0.2	67	5.9	9.3

从表 3 可以看出,本算法在这 5 个数据集上取得了与在入侵检测数据集上同样的效果,尤其是在 W1a 数据

集上,本简化法在取得 95%约简率的同时,却没有任何泛化性能的损失.另一方面,在相同约简率下,除了在 Mushroom 数据集上两种简化法都没有造成泛化性能下降以外,在其他 4 个数据集上,本简化法所获得精简的 SVM 的分类精度损失都远远低于 Scholkopf 方法所获得的结果,这进一步验证了本简化法的有效性.

4 结束语

针对 SVM 分类速度慢的问题,本文提出了一种新颖的基于核聚类的 SVM 简化方法.与原有方法相比,此简化法具有如下优势:(1) 概念简单,约简向量具有明确的物理意义.(2). 在简化过程中只需求解线性代数问题,从而解决了现有简化法求解过程中存在的局部最小值问题.(3) 本简化法具有较大的约简率和较低的分类精度损失.在入侵检测数据和其他数据集上的实验结果表明,本文所提出的 SVM 简化法在基本保持 SVM 泛化性能的同时,有效地削减了支持向量的数量,提高了 SVM 的分类速度,从而增强了 SVM 这种优秀学习方法的应用范围.在接下来的工作中,我们将进一步完善此约简方法,使其适用于任意核函数.

References:

[1] Vapnik V. Statistical Learning Theory. New York: John Wiley & Sons, 1998.

[2] Bian ZQ, Zhang XG, *et al*. Pattern Recognition. 2nd ed., Beijing: Tsinghua University Press, 2000. 284−303 (in Chinese).

[3] Downs T, Gates KE, Masters A. Exact simplification of support vector solutions. Journal of Machine Learning Research, 2001,2: 293−297.

[4] Scholkopf B, Mika S, Burges C, Knirsch P, Muller KR, Ratsch G, Smola A. Input space versus feature space in kernel-based methods. IEEE Trans. on Neural Networks, 1999,10(5):1000−1017.

[5] Scholkopf B, Knirsch P, Smola A, Burges C. Fast approximation of support vector kernel expansions, and an interpretation of clustering as approximation in feature spaces. In: Levi P, Schanz M, Ahlers RJ, May F, eds. Mustererkennung. Berlin: Springer-Verlag, 1998. 124−132.

[6] Mika S, Scholkopf B, Smola A, Muller KR, Scholz M, Ratsch G. Kernel PCA and de-noising in feature spaces. In: Kearns MJ, ed. Advances in Neural Information Processing Systems 11. San Mateo: Morgan Kaufmann Publishers, 1998. 536−542.

[7] Burges CJC. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 1998,2(2): 121−167.

[8] Li XL, Liu JM, Shi ZZ. A Chinese Web page classifier based on support vector machine and unsupervised clustering. Chinese Journal of Computers, 2001,24(1):62−68 (in Chinese with English abstract).

[9] Kwok JT, Tsang IW. The pre-image problem in kernel methods. IEEE Trans. on Neural Networks, 2004,15(6):1517−1525.

[10] Williams CKI. On a connection between kernel PCA and metric multidimensional scaling. Machine Learning, 2002,46(1/3):11−19.

[11] Gower JC. Adding a point to vector diagrams in multivariate analysis. Biometrika, 1968,55(3):582−585.

[12] Chang CC, Lin CJ. LIBSVM: A library for support vector machines. 2001. http://www.csie.ntu.edu.tw/~cjlin/libsvm

[13] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[14] Murphy PM, Aha DW. UCI repository of machine learning databases. Irvine, 1994. http://www.ics.uci.edu/~mlearn/MLRepository.html

[15] Hsu CW, Chang CC, Lin CJ. A practical guide to support vector classification [BB/OL]. 2003-08-10/2004-11-10, http://www.csie.ntu.edu.tw/-cjlin/papers/guide/guide.pdf

附中文参考文献:

[2] 边肇祺,张学工,等.模式识别.第 2 版,北京:清华大学出版社,2000.284−303.

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

曾志强(1971－),男,福建厦门人,博士生,主要研究领域为模式识别,机器学习.

高济(1946－),男,教授,博士生导师,主要研究领域为人工智能,Agent 技术,网格计算.