一种并行结构化支持向量机次梯度投影算法

郭丽娜 杨 明 涂金金
(南京师范大学计算机科学与技术学院 南京 210046)

摘 要 支持向量机的次梯度投影算法是解决支持向量机优化求解问题的一种简单有效的迭代算法。该算法通过梯度下降和投影两个步骤的多轮迭代,找到两类最大间隔的分类面。针对该算法忽略了对寻找分类面同样有指导意义的样本分布信息这一问题,在分类器设计中融入结构信息,并且采用MapReduce并行计算框架,提出了一种并行结构化支持向量机的次梯度投影算法,该算法能够充分利用集群的计算和存储能力,适用于海量数据的优化问题。在NASA的两个软件模块缺陷度量数据集CM1和PC1上的实验结果表明,该算法能够加快收敛速度,提高分类性能,有效地解决海量数据的优化求解问题。

关键词 结构化支持向量机,并行,MapReduce
中图法分类号 TP181 **文献标识码** A

Parallel Primal Estimated Sub-GrAdient Solver for Structural SVM

GUO Li-na YANG Ming TU Jin-jin
(School of Computer Science and Technology, Nanjing Normal University, Nanjing 210046, China)

Abstract Primal estimated sub-GrAdient solver for SVM (Pegasos) is a simple and effective iterative algorithm for solving the optimization problem of Support Vector Machine. The method alternates between stochastic gradient descent steps and projection steps to find a hyper-plane that can separate two classes of samples with the maximal margin. But it neglects the data distributions which are also vital for an optimal classifier. We developed a novel algorithm, termed as Parallel Primal Estimated sub-GrAdient Solver for Structural SVM (PSPegasos) by embedding the structural information into the SVM and using the parallel computing framework: MapReduce. This algorithm can take full advantage of the computing and storage capacity of the computer cluster, and be applicable to the optimization problem of the massive data. The algorithm was used to two NASA software module datasets CM1 and PC1, and the experimental results show that the algorithm can accelerate the convergence speed, improve the classification performance and be an effective solution to the optimization problem of the massive data.

Keywords Structural support vector machine, Parallel, MapReduce

1 引言

支持向量机(SVM)[1]由于其完备的理论基础和卓越的学习性能,成为机器学习中的研究热点,并且在很多领域都得到成功应用。但是随着需要处理的数据量的不断增大,计算内存和运算时间成为SVM求解瓶颈,很大程度上制约了SVM的发展。针对这些问题,人们提出了并行SVM的解决方法,如使用多个SVM分类器的组合来解决大规模数据分类问题的方法[2]、分层SVM学习方法[3]等。

进一步,MapReduce[4]并行计算框架近年来受到了广泛的关注和研究,它于2004年由Google公司提出,特别适用于处理大规模海量数据。Map和Reduce是其两大基本操作。用户通过Map函数处理key/value(键/值)对,产生一系列中间key/value对,在Reduce函数中将具有相同key的中间key/value对归并得到最终结果。在MapReduce框架下,并行SVM研究也取得一些进展。Zhu等人提出了一种并行SVM算法,运用内点法求解SVM时,通过基于行的近似矩阵分解来减少内存的使用[5]。文献[6]中,将基于MapReduce的并行编程模型与增量学习的方法结合使用。在使用随机次梯度投影法求解SVM时,Zhao也提出了基于MapReduce模型的并行方法的思路[7]。

并行SVM的目标仍是寻找以最大间隔将两类分开的最优分类面。与传统的SVM一样,更多关注的是类之间的分散性,而忽略了对最终的分类面同样具有指导意义的各类的分布信息。虽然近年来也相继提出了很多考虑样本结构信息的大间隔分类器算法,如最小最大概率机[8]、最大最小间隔[9]、结构大间隔机[10]、结构支持向量机[11]等,但是考虑样本的结构信息的并行SVM的研究却很少见。为了有效地利

到稿日期:2013-05-20 返修日期:2013-08-02 本文受国家自然科学基金(61272222,61003116),江苏省自然科学基金重点重大专项(BK2011005),江苏省自然科学基金(BK2011782),江苏省普通高校研究生科研创新计划项目(CXLX12_0415)资助。
郭丽娜(1989-),女,硕士生,主要研究方向为机器学习、模式识别,E-mail:guolinagogo@126.com;杨 明(1964-),男,博士,教授,主要研究方向为机器学习、模式识别;涂金金(1988-),男,硕士生,主要研究方向为机器学习、模式识别。

用样本的结构信息，获得更加合理的分类决策面，提高分类精度，并且能够适用于处理海量数据分类问题，本文提出了一种并行结构化支持向量机的次梯度投影算法。在支持向量机中融入结构信息，基于 MapReduce 框架，对次梯度投影算法进行有效的并行。在 NASA 软件模块缺陷度量的两个数据集上的实验结果证明了新提出的算法的有效性。

2 支持向量机的次梯度投影算法(Pegasos)

支持向量机的次梯度投影算法(Pegasos)是 Shai 等人提出的一种求解支持向量机的简单的迭代算法[12]。该算法在多次迭代中交替地执行随机梯度下降和投影步骤。每次迭代中，都会从整体的训练样本中抽取一定数量的样本来计算每一轮迭代的次梯度。

为了方便描述，记给定训练集 $S=\{(\boldsymbol{x}_i,y_i)\}_{i=1}^m$，其中 $\boldsymbol{x}_i\in R^n$，$y_i\in\{+1,-1\}$，支持向量机问题为：

$$\min_{\boldsymbol{w}}\frac{\lambda}{2}\|\boldsymbol{w}\|^2+\frac{1}{m}\sum_{(\boldsymbol{x},y)\in S}\ell(\boldsymbol{w};(\boldsymbol{x},y)) \tag{1}$$

其中，$\ell(\boldsymbol{w};(\boldsymbol{x},y))=\max\{0,1-y\langle\boldsymbol{w},\boldsymbol{x}\rangle\}$。将式(1)记为 $f(\boldsymbol{w})$，如果有 $f(\hat{\boldsymbol{w}})\leqslant\min_{\boldsymbol{w}}f(\boldsymbol{w})+\varepsilon$，则 $\hat{\boldsymbol{w}}$ 为精确度 ε 范围内的解。Pegasos 算法描述如下：

算法 1 Pegasos

输入：训练数据集 S，迭代次数 T，每轮迭代选取的样本个数 k。

输出：分类超平面的法向量 w。

主要步骤：

1. 初始化向量 w。任取一向量 $\mathbf{w}_1$，并要求其满足 $\|\mathbf{w}_1\|\leqslant1/\sqrt{\lambda}$。

2. For t=1 to T

2.1 从训练集 S 中选取 k 个样本的子集 $A_t\in S$，并用以下的目标函数来代替原来的目标函数，

$$\min_{\mathbf{w}}\frac{\lambda}{2}\|\mathbf{w}\|^2+\frac{1}{k}\sum_{(\mathbf{x},y)\in A_t}\max\{0,1-y\langle\mathbf{w},\mathbf{x}\rangle\}$$

2.2 确定梯度下降法的学习率为 $\eta_t=\frac{1}{\lambda t}$。

2.3 将 A_t 中使用 $\mathbf{w}_t$ 判断当前损失非零的样本组成一个新的子集 A_t^+，即 $A_t^+=\{(\mathbf{x}_i,y_i)\in A_t:y_i\langle\mathbf{w}_t,\mathbf{x}_i\rangle<1\}$，目标函数的次梯度方向可以表示为 $\nabla_t=\lambda\mathbf{w}_t-\frac{1}{|A_t|}\sum_{(\mathbf{x}_i,y_i)\in A_t^+}y_i\mathbf{x}_i$。

2.4 更新：

$$\mathbf{w}_{t+\frac{1}{2}}=\mathbf{w}_t-\eta_t\nabla_t=(1-\eta_t\lambda)\mathbf{w}_t+\frac{\eta_t}{k}\sum_{(\mathbf{x}_i,y_i)\in A_t^+}y_i\mathbf{x}_i$$

2.5 投影步骤：

$$\mathbf{w}_{t+1}=\min\{1,\frac{1/\sqrt{\lambda}}{\|\mathbf{w}_{t+\frac{1}{2}}\|}\}\mathbf{w}_{t+\frac{1}{2}}。$$

3. 得到最终结果 $\mathbf{w}_{T+1}$。

3 并行结构化支持向量机次梯度投影算法(PSPegasos)

3.1 结构化支持向量机的次梯度投影算法(SPegasos)

近年来，出现很多考虑了样本结构信息的大间隔分类器的研究。这样的分类器对数据分布结构敏感。数据的协方差矩阵包含着统计意义上的数据产生趋势，因此将训练样本的协方差矩阵作为结构信息加入到目标函数中，可以在最大化间隔的同时，最小化类内紧性。

结构化支持向量机的模型可以写成：

$$\min_{\boldsymbol{w}}\frac{\lambda_1}{2}\|\boldsymbol{w}\|^2+\frac{1}{m}\sum_{(\boldsymbol{x},y)\in S}\ell(\boldsymbol{w};(\boldsymbol{x},y))+\frac{\lambda_2}{2}\boldsymbol{w}^{\mathrm{T}}\Sigma\boldsymbol{w} \tag{2}$$

其中，$\ell(\boldsymbol{w};(\boldsymbol{x},y))=\max\{0,1-y\langle\boldsymbol{w},\boldsymbol{x}\rangle\}$，$\Sigma$ 为样本的协方差矩阵。λ_1 和 λ_2 为平衡各项之间的正则化因子，$\lambda_1,\lambda_2>0$。

Pegasos 算法中，每次迭代之后的 $\boldsymbol{w}$ 都会投影到集合 $B=\{\boldsymbol{w}:\|\boldsymbol{w}\|\leqslant1/\sqrt{\lambda}\}$ 之内。而 SPegasos 算法中，由于结构信息的加入，$\boldsymbol{w}$ 的投影范围也发生了变化，原本的投影范围不再适用。见命题 1：

命题 1 结构化支持向量机的最优化解满足约束条件：$\boldsymbol{w}^{\mathrm{T}}(\lambda_1 I+\lambda_2\Sigma)\boldsymbol{w}\leqslant1$，$I$ 为单位矩阵。

证明：将式(2)的最优化解表示为 $\boldsymbol{w}^*$，式(2)所描述的优化问题的对偶问题可表示为：

$$\max_{\boldsymbol{\alpha}\in[0,1/m]^m}\sum_{i=1}^m\alpha_i-\frac{1}{2}\sum_{i=1}^m\sum_{j=1}^m\alpha_i\alpha_j y_i y_j\boldsymbol{x}_i^{\mathrm{T}}(\lambda_1 I+\lambda_2\Sigma)^{-1}\boldsymbol{x}_j$$

令 $\boldsymbol{\alpha}^*$ 为对偶问题的解。

由强对偶定理可知，原问题的目标函数值等于对偶问题的目标函数值。另有 $(\lambda_1 I+\lambda_2\Sigma)\boldsymbol{w}^*=\sum_{i=1}^m\alpha_i^*y_i\boldsymbol{x}_i$，即可得以下式子：

$$\frac{\lambda_1}{2}\|\boldsymbol{w}^*\|^2+\frac{1}{m}\sum_{i=1}^m\max\{0,1-y_i\langle\boldsymbol{w}^*,\boldsymbol{x}_i\rangle\}+\frac{\lambda_2}{2}\boldsymbol{w}^{*\mathrm{T}}\Sigma\boldsymbol{w}^*$$
$$=\sum_{i=1}^m\alpha_i^*-\frac{1}{2}\boldsymbol{w}^{*\mathrm{T}}(\lambda_1 I+\lambda_2\Sigma)\boldsymbol{w}^*$$

即

$$\boldsymbol{w}^{*\mathrm{T}}(\lambda_1 I+\lambda_2\Sigma)\boldsymbol{w}^*=\sum_{i=1}^m\alpha_i^*-\frac{1}{m}\sum_{i=1}^m\max\{0,1-y_i\langle\boldsymbol{w}^*,\boldsymbol{x}_i\rangle\}$$

又有 $\sum_{i=1}^m\alpha_i^*\leqslant1$，$\frac{1}{m}\sum_{i=1}^m\max\{0,1-y_i\langle\boldsymbol{w}^*,\boldsymbol{x}_i\rangle\}\geqslant0$，所以 $\boldsymbol{w}^{*\mathrm{T}}(\lambda_1 I+\lambda_2\Sigma)\boldsymbol{w}^*\leqslant1$。证毕。

由命题 1 可知，SPegasos 算法的最优解在集合 $B=\{\boldsymbol{w}:\boldsymbol{w}^{\mathrm{T}}(\lambda_1 I+\lambda_2\Sigma)\boldsymbol{w}\leqslant1\}$ 中，所以算法在每轮迭代中梯度下降步骤完成之后，将 $\boldsymbol{w}$ 投影到 B 中，即将 $\boldsymbol{w}$ 乘以 $\min\{1,1/\sqrt{\boldsymbol{w}^{\mathrm{T}}(\lambda_1 I+\lambda_2\Sigma)\boldsymbol{w}}\}$，能够使 $\boldsymbol{w}$ 更加逼近最优解。SPegasos 算法的具体描述如下：

算法 2 SPegasos

输入：训练数据集 S，迭代次数 T，每轮迭代选取的样本个数 k。

输出：分类超平面的法向量 w。

主要步骤：

1. 计算出样本的协方差矩阵 Σ；

2. 初始化向量 w。任取向量 $\mathbf{w}_1$，使其满足 $\mathbf{w}^{\mathrm{T}}(\lambda_1\Lambda+\lambda_2\Sigma)\mathbf{w}\leqslant1$。

3. For t=1 to T

3.1 从训练集 S 中选取样本个数为 k 的子集 $A_t\in S$，并用以下的目标函数来代替原来的目标函数，$\min_{\mathbf{w}}\frac{\lambda_1}{2}\|\mathbf{w}\|^2+\frac{1}{k}\sum_{(\mathbf{x},y)\in A_t}\max\{0,1-y\langle\mathbf{w},\mathbf{x}\rangle\}+\frac{\lambda_2}{2}\mathbf{w}^{\mathrm{T}}\Sigma\mathbf{w}$。

3.2 确定梯度下降法的学习率为 $\eta_t=\frac{1}{\lambda t}$。

3.3 将 A_t 中使用 $\mathbf{w}_t$ 判断当前损失非零的样本组成一个新的子集 A_t^+，即 $A_t^+=\{(\mathbf{x}_i,y_i)\in A_t:y_i\langle\mathbf{w}_t,\mathbf{x}_i\rangle<1\}$，目标函数的次梯度方向可以表示为 $\nabla_t=\lambda_1\mathbf{w}_t-\frac{1}{|A_t|}\sum_{(\mathbf{x}_i,y_i)\in A_t^+}y_i\mathbf{x}_i+\lambda_2\Sigma\mathbf{w}_t$。

3.4 更新：

$$\mathbf{w}_{t+\frac{1}{2}}=\mathbf{w}_t-\eta_t\nabla_t=(1-\eta_t\lambda_1)\mathbf{w}_t+\frac{\eta_t}{k}\sum_{(\mathbf{x}_i,y_i)\in A_t^+}y_i\mathbf{x}_i-\eta_t\lambda_2\Sigma\mathbf{w}_t$$

3.5 投影步骤：

$$\mathbf{w}_{t+1}=\min\{1,1/\sqrt{\mathbf{w}_{t+\frac{1}{2}}^{T}(\lambda_1\Lambda+\lambda_2\Sigma)\mathbf{w}_{t+\frac{1}{2}}}\}\mathbf{w}_{t+\frac{1}{2}}。$$

4. 得到最终结果 $\mathbf{w}_{T+1}$。

3.2 并行 SPegasos(PSPegasos)

本文使用 MapReduce 框架实现 SPegasos 算法的并行化。在此框架下，PSPegasos 算法可以分为获得样本的结构化信息(即计算样本的协方差矩阵)和次梯度投影迭代两个阶段。计算样本的协方差矩阵以及次梯度投影的每一个迭代都包含着一个独立的 MapReduce 任务，

首先，获得样本的结构化信息。在 MapReduce 框架下，训练样本分散地放在各个数据节点上，该阶段的任务就是求得整个训练集样本的协方差矩阵。为了方便描述，记给定训练集 $S=\{(\boldsymbol{x}_i,y_i)\}_{i=1}^{m}$，其中 $\boldsymbol{x}_i\in R^n$，$y_i\in\{+1,-1\}$。将 S 分成 N 个子集，记为 $S_i=\{(\boldsymbol{x}_j,y_j)\}_{j=1}^{N_i}$，$i=1,\cdots,N$，$y_j\in\{+1,-1\}$。记 $\Sigma_i=\sum\limits_{\boldsymbol{x}_j\in S_i}\boldsymbol{x}_j\boldsymbol{x}_j^{T}$，$\mu_i=\sum\limits_{\boldsymbol{x}_j\in S_i}\boldsymbol{x}_j$，$i=1,\cdots,N$。有如下命题：

命题 2 训练集 S 上样本协方差矩阵 Σ 可分解到各个子集求解，即

$$\Sigma=\frac{1}{m}\sum_{i=1}^{N}\Sigma_i-\frac{1}{m^2}\sum_{i=1}^{N}\mu_i\sum_{i=1}^{N}\mu_i^{T}$$

其中，$m=\sum\limits_{i=1}^{N}N_i$。

证明：训练集 S 的协方差矩阵可以表示为 $\Sigma=\frac{1}{m}\sum\limits_{i=1}^{m}(\boldsymbol{x}_i-\mu)(\boldsymbol{x}_i-\mu)^{T}$，其中 $\mu=\frac{1}{m}\sum\limits_{i=1}^{m}\boldsymbol{x}_i$。对 Σ 进行化简可得到：

$$\begin{aligned}\Sigma&=\frac{1}{m}\sum_{i=1}^{m}\boldsymbol{x}_i\boldsymbol{x}_i^{T}-\frac{1}{m}\sum_{i=1}^{m}\boldsymbol{x}_i\frac{1}{m}\sum_{j=1}^{m}\boldsymbol{x}_j\\&=\frac{1}{\sum\limits_{i=1}^{N}N_i}\sum_{i=1}^{N}\sum_{\boldsymbol{x}_j\in S_i}\boldsymbol{x}_j\boldsymbol{x}_j^{T}-\frac{1}{\sum\limits_{i=1}^{N}N_i}\sum_{i=1}^{N}\sum_{\boldsymbol{x}_j\in S_i}\boldsymbol{x}_j\frac{1}{\sum\limits_{i=1}^{N}N_i}\sum_{k=1}^{N}\sum_{\boldsymbol{x}_j\in S_k}\boldsymbol{x}_j^{T}\\&=\frac{1}{m}\sum_{i=1}^{N}\Sigma_i-\frac{1}{m^2}\sum_{i=1}^{N}\mu_i\sum_{i=1}^{N}\mu_i^{T}\end{aligned}$$

其中，$m=\sum\limits_{i=1}^{N}N_i$。证毕。

由命题 2 可知，求解整个训练集样本的协方差矩阵可用一个 MapReduce 任务完成。在 Map 阶段：依次扫描当前节点 i 上的样本，求得当前节点样本个数 N_i、$\Sigma_i=\sum\limits_{\boldsymbol{x}_j\in S_i}\boldsymbol{x}_j\boldsymbol{x}_j^{T}$、$\mu_i=\sum\limits_{\boldsymbol{x}_j\in S_i}\boldsymbol{x}_j$；Reduce 阶段：汇总 N 个节点的 N_i、Σ_i、μ_i，$i=1,\cdots,N$。根据命题 2，得到整个训练集样本的协方差矩阵。

接下来为次梯度投影迭代阶段，该阶段为算法的核心部分，每次迭代作为一个单独的 MapReduce 任务。第 t 次迭代中，Map 和 Reduce 两阶段的具体算法步骤描述如下：

//Map：

输入：$\mathbf{w}_t$，节点个数 M，每轮迭代随机取出的样本的个数 k；

输出：当前节点求得的 $\mathbf{v}_j=\sum\limits_{(\mathbf{x}_i,y_i)\in A_t^{+}}y_i\mathbf{x}_i$，其中 $A_t^{+}=\{(\mathbf{x}_i,y_i)\in A_t:y_i\langle\mathbf{w}_t,\mathbf{x}_i\rangle<1\}$

主要步骤：

1. 随机抽取 k/M 个样本

2. 定义零向量 $\mathbf{v}_j\in R^n$

3. For i=1 to k/M

3.1 If $y_i*\mathbf{w}_t^{T}\mathbf{x}_i<1$, then $\mathbf{v}_j=\mathbf{v}_j+y_i*\mathbf{x}_i$

4. End

//Reduce：

输入：$\mathbf{w}_t$，节点个数 M，每轮迭代随机取出的样本的个数 k，M 个节点的 Map 输出 $\mathbf{v}_j$，j=1,…,M，正则化参数 λ_1，λ_2，当前迭代次数 t；

输出：$\mathbf{w}_{t+1}$；

主要步骤：

1. 计算出 $\mathbf{v}=\sum\limits_{j=1}^{M}\mathbf{v}_j$

2. $\eta_t=\lambda_1*t$

3. $\mathbf{w}_{t+\frac{1}{2}}=(1-\eta_t\lambda_1)\mathbf{w}_t+\frac{\eta_t}{k}\mathbf{v}-\eta_t\lambda_2\Sigma\mathbf{w}_t$

4. $\mathbf{w}_{t+1}=\min\{1,1/\sqrt{\mathbf{w}_{t+\frac{1}{2}}^{T}(\lambda_1\Lambda+\lambda_2\Sigma)\mathbf{w}_{t+\frac{1}{2}}}\}\mathbf{w}_{t+\frac{1}{2}}$

4 实验

为了进一步测试新提出的算法的性能，在 NASA 的软件模块缺陷度量数据集 CM1 和 PC1 上进行了实验。

4.1 实验数据集描述

CM1 是用 C 语言编写的将近 20k 行代码的一个科学仪器中的应用程序的软件度量数据集，该数据集有 505 个模块(48 个有缺陷模块，457 个无缺陷模块)和 37 个模块集的度量属性。PC1 也是 C 语言编写的将近 40k 行代码的一个不再运行的卫星飞行软件的软件度量数据集，该数据集有 1107 个模块(76 个有缺陷模块，1031 个无缺陷模块)和 37 个模块集的度量属性。

4.2 样本预处理

将每个数据集进行归一化处理，根据 Shai 给出的 Pegasos代码的测试用例[1)]，样本的属性值普遍较小。为了获得更好的实验效果，本文也采取类似设置，将样本归一化到[0，0.5]。从上一节数据集描述可以看到，实验中的数据集都属于不平衡数据集，所以，首先通过随机过抽样，使正负类样本达到相对平衡。具体操作如下：

首先，将各个数据集以 2∶1 的比例划分为训练集和测试集两个部分。对各个训练集的正类样本进行过抽样，即复制正类样本，最终选择效果最好的复制比例。CM1 的训练集(CM1_train)包含 32 个正类样本和 305 个负类样本。将其中的全部正类样本复制 7 份(224 个)与原先的负类样本构成一个新的训练集(CM1_b)，共 529 个样本。PC1 的训练集(PC1_train)包含 50 个正类样本和 688 个负类样本。将其中的正类全部样本复制 13 份(650 个)与原先的负类样本构成一个新的训练集(PC1_b)，共 1338 个样本。具体描述见表 1。

表 1 数据集 CM1 和 PC1 的基本信息

数据集	度量属性数	数据集划分情况		正类样本数	负类样本数
CM1	37	训练集	CM1_train	32	305
		训练集(预处理后)	CM1_b	224	305
		测试集	CM1_test	16	152
PC1	37	训练集	PC1_train	50	688
		训练集(预处理后)	PC1_b	650	688
		测试集	PC1_test	26	343

接下来，为了验证 PSPegasos 算法处理海量数据集的性能，将 CM1_b 和 PC1_b 两个训练集通过不同的倍数复制，分别构造出 10M、100M、1G 左右大小的 3 个人工训练集。具体描述见表 2。

1) http://www.cs.huji.ac.il/~shais/code/index.html

表 2　人工数据集的基本信息

训练集名称	复制倍数	数据集大小
CM1_b	1	103.8k
CM1_b_10M	100	10.4M
CM1_b_100M	1000	103.8M
CM1_b_1G	10000	0.99G
PC1_b	1	256.9k
PC1_b_10M	40	10.3M
PC1_b_100M	400	102.8M
PC1_b_1G	4000	1G

4.3　实验设置

本文实验分别在个人计算机和云计算平台上执行。配置如下：个人计算机主频 3.29GHZ，内存 1.84GB；云计算平台共有 21 台服务器（1 个 master 节点、20 个 slave 节点），每个节点配置相同。节点的处理器为 Intel(R) Xeon(R) CPU E5620，主频为 2.40GHZ，操作系统为 64 位 Debian Linux。Hadoop 平台版本为 hadoop-0.20.2。本文除了使用正确率这一性能度量指标之外，还使用 TPR、TNR 和 GM。GM 是用于不平衡数据分类问题的一种有效的性能度量指标，GM 值比较大就表示这个分类器对两类都有很好的预测性能，没有偏向某一类。根据表 3 所列的二阶混淆矩阵，得到 TPR、TNR、GM，见式(3)－式(5)。

$$TPR=\frac{TP}{TP+FN} \tag{3}$$

$$TNR=\frac{TN}{TN+FP} \tag{4}$$

$$GM=\sqrt{(TPR)\cdot(TNR)} \tag{5}$$

表 3　二阶混淆矩阵

		预测	
		正类	负类
实际	正类	TP	FN
	负类	FP	TN

4.4　实验结果分析

在第一个实验中，实验参数设置如下：Pegasos 算法中 $T=100000$，$k=1$，lambda = 0.001，SPegasos 算法中 $T=100000$，$k=1$，lambda1=0.001，lambda2=0.01。分别在 CM1_b、PC1_b 两个训练集上训练，在 CM1_test、PC1_test 上测试，实验结果如表 4 所列。

表 4　Pegasos 和 SPegasos 算法在两个数据集上的测试结果

数据集	训练算法	Accuracy (%)	Tpr (%)	Tnr (%)	GM (%)
CM1	Pegasos	79.76	83.13	79.34	81.14
	SPegasos	81.48	83.75	79.80	81.72
PC1	Pegasos	72.3	74.62	72.22	73.38
	SPegasos	73.4	75.38	73.18	74.26

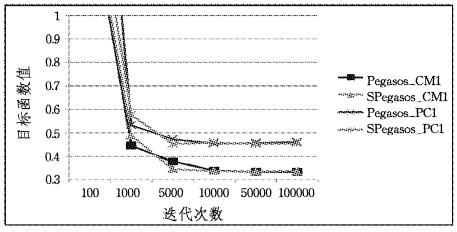

图 1　CM1 和 PC1 两个数据集上，结构信息的加入对 Pegasos 算法的收敛性能的影响

从上述的实验结果可以看出，在支持向量机次梯度投影中加入结构信息后，整个测试集上的测试正确率、正类样本的分类正确率、GM 值都有所提高。由图 1 可以看出，结构信息的加入不仅使分类精度有所改进，还能够加快收敛速度。

下一个实验中，采用 SPegasos 算法，分别在 CM1_b、PC1_b 上训练，在 CM1_test、PC1_test 上测试。可以发现当 kT 固定在一个合适的值，并且 k 和 T 在一定范围内变动时，算法的收敛率、分类效果几乎保持不变。

表 5　当 kT 固定时，k、T 各自的取值对 SPegasos 算法性能的影响

数据集	T	k	GM(%)
CM1	10^5	1	81.815
	10^4	10	81.896
	10^3	10^2	81.810
	10^2	10^3	81.048
PC1	10^5	1	74.2595
	10^4	10	74.028
	10^3	10^2	74.39
	10^2	10^3	73.6053

正是基于这样的结论，提出了 PSPegasos 算法，该算法中选取较小的 T 值、较大的 k 值，分散到多个节点执行。在接下来的实验中将 SPegasos 和 PSPegasos 中的实验参数都设置为 $T=100$，$k=1000$，lambda1=0.001，lambda2=0.01；本实验中一共使用 20 个 map 节点，因此每个 map 节点的 k 设置为 50；从表 6 实验结果中可以看到，PSPegasos 算法由于多次迭代花费在 hadoop 启动集群上的时间较长，即使在数据集比较小时，也需要花费较多的时间，不能表现出它的优势。但当数据集规模不断增大时，花费的时间并没有大幅增加，因而能够有效地处理海量数据集的分类问题。

表 6　Pegasos 与 PSPegasos 算法在分类时间上的比较

训练集	测试集	SPegasos		PSPegasos	
		GM(%)	Time(ms)	GM(%)	Time(ms)
CM1_b _10M	CM1_test	81.7	4005	81.7	1786664
CM1_b_100M		81.8	709283	81.8	1808134
CM1_b_1G		—	—	81.7	2110350
PC1_b_10M	PC1_test	74.01	4235	74.2	1719472
PC1_b_100M		74.1	758328	74.1	1932817
PC1_b_1G		—	—	74.1	1957384

结束语　本文提出的 PSPegasos 算法与 Pegasos 算法相比，通过结构信息的加入，在分类精度上有所改进；通过基于 MapReduce 的并行框架，能够借助集群的力量，适用于海量数据集优化求解问题。接下来，可以在该算法中核函数的使用以及选择更加适用的迭代式 MapReduce 执行框架等方面进行进一步的研究。

参 考 文 献

[1] Cortes C, Vapnik V. Support vector networks [J]. Machine Learning, 1995, 20(2): 273-295

[2] Collobert R, Bengio S, Bengio Y. A Parallel Mixture of SVMs for Very Large Scale Problems[J]. Neural computation, 2002, 14(5): 1105-1114

[3] Graf H P, Cosatto E, Bottou L, et al. Parallel Support Vector Machines[C] // The Cascade SVM: Proceedings of the Eighteenth Annual Conference on Neural Information Processing Systems, 2005. Vancouver, Canada: MIT Press, 2005: 521-528

（下转第 135 页）

开较强的组合火力而受到较大的伤害值。一般来说，模糊测度越大，交互效果越明显，大炮之间的攻击效率较无交互的越高，攻方在寻路中遭受的伤害较原来直接叠加的伤害也越大。

表5 不同模糊测度值结果比较

组别	超可加	可加	次可加
模糊测度	λ=0.3	λ=0	λ=-0.3
	μ(A)=2	μ(A)=2	μ(A)=2
	μ(B)=3	μ(B)=3	μ(B)=3
	μ(C)=4	μ(C)=4	μ(C)=4
	μ(A∪B)=6.8	μ(A∪B)=5	μ(A∪B)=3.2
	μ(A∪C)=8.4	μ(A∪C)=6	μ(A∪C)=3.6
	μ(B∪C)=10.6	μ(B∪C)=7	μ(B∪C)=3.4
	μ(A∪B∪C)=16.8	μ(A∪B∪C)=9	μ(A∪B∪C)=1.2
伤害值	293	218	143

结束语 本文选取RTS游戏中一种典型的攻防场景，从攻方的角度考虑，在守方动态变换火力位置的情况下，对比人工势场法寻路、A* 算法和Dijkstra算法寻路的耗时和攻方所受伤害值，并在势场法寻路中引入模糊测度，考虑到防守单元之间的交互作用，计算不同模糊测度下势场寻路的伤害值。从实验结果看，Dijkstra算法寻得最短路径，寻路所受伤害值相对较小，但寻路所耗时间较大；A* 算法在障碍物较少、转角少的地图中寻路较快，但不适用于复杂的多障碍的地图；人工势场法寻路较Dijkstra算法寻路效率高，时间明显缩短，寻得的路径较平滑，但势场法只考虑障碍点坐标，只有进入大炮射击范围内才计算斥力，因此在寻路过程中遭受的伤害值比前者稍高。在势场法寻路中引入模糊测度，模糊测度值越大，伤害值越高，因此势场法寻路与真实的游戏场景更为接近。未来的工作是将单元之间的交互作用直接在势场的斥力公式中表示出来，以在各种交互模式中高效地完成动态寻路任务。

参 考 文 献

[1] 刘天白，张舒白. 机器学习技术在游戏中的应用研究——解决鼠标轨迹识别问题[J]. 电脑知识与技术，2011，7(13)：3100-3102

[2] 杨佩，王皓，罗文杰. 等. HUNTBot—第一人称射击游戏中NPC的结构设计[J]. 计算机科学，2008，35(11)：290-292

[3] Huo P，Shiu S C-K，Wang H，et al. A neural evolutionary model for case-based planning in real time strategy games[C]// Next-Generation Applied Intelligence 2009. Berlin Heidelberg，2009：291-300

[4] Huo P，Shiu S C-K，Wang H，et al. Application and comparison of particle swarm optimization and genetic algorithm in strategy defense games[C]// Fifth International Conference on Natural Computation. IEEE，2009：387-392

[5] Tong Xiao-lei，Li Yan，Li Wen-liang. Optimization Methods For Resources Allocation In Real-Time Strategy Games[C]// Proceedings of the 2011 International Conference on Machine Learning and Cybernetics. Guilin，2011，2：507-513

[6] Peter H F N，Li Y J，Shiu S C K. Unit Formation Planning in RTS game by using Potential Field and Fuzzy Integral[C]// 2011 IEEE International Conference on Fuzzy Systems (FUZZ-Y). Taipei，2011：178-184

[7] Li Ying-jie，Li Yan，Simon C K S，et al. RTS game strategy evaluation using extreme learning machine[J]. Soft Computing，2012，16(9)：1627-1637

[8] Hart P E，Nilsson N J，Raphael B. A formal basis for the heuristic determination of minimum cost paths in graphs[J]. IEEE Trans. Syst. Sci. and Cybernetics，SSC，1968，4(2)：100-107

[9] Dijkstra E W. A note on two problems in connexion with graphs[J]. Numerische Mathematik，1959，1：269-271

[10] Khatib O. Real-Time Obstacle Avoidance For Manipulators and Mobile Robots[J]. Int. J. Robotics Res，1986，5(1)：90-98

[11] Hagelback J，Johansson S. Using multi-agent potential fields in real-time strategy games[C]// Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems. 2008：631-638

[12] Sugeno M. Theory of fuzzy integrals and its applications[D]. Tokyo：Tokyo Institute of Technology，1974

[13] 胡志华. 基于离散人工势场的移动机器人动态路径规划研究[D]. 长沙：中南大学，2009

[14] Peters J，Han L，Ramanna S. The Choquet integral in a rough software cost decision system[M]// Fuzzy Measures and Integrals：Theory and Applications，Studies in fuzziness and soft computing，1999

(上接第130页)

[4] Dean J，Ghemawat S. MapReduce：simplified data processing on large clusters[J]. Communications of the ACM，2008，51(1)：107-113

[5] Chang E Y. Foundations of Large-Scale Multimedia Information Management and Retrieval[M]. Berlin，Heidelberg：Springer，2011：213-230

[6] Zhao J，Liang Z，Yang Y. Parallelized incremental support vector machines based on MapReduce and Bagging technique[C]// Proceedings of the International Conference on Information Science and Technology，2012. New York：IEEE，2012：297-301

[7] Zhao Z. Parallel Pegasos for Mahout [OL]. http://wenku.baidu.com/view/0e8d6d639b6648d7c1c74661.html，2012

[8] Lanckriet G R，Ghaoui L E，Bhattacharyya C，et al. A Robust Minimax Approach to Classification[J]. Journal of Machine Learning Research，2002，3(3)：555-582

[9] Huang K，Yang H，King I，et al. Learning large margin classifiers locally and globally[C]// Proceedings of the twenty-first international conference on Machine learning，2004. New York：ACM，2004：51

[10] Yeung D S，Wang D，Ng W W，et al. Structured Large Margin Machines：Sensitive to Data Distributions[J]. Machine Learning，2007，68：171-200

[11] Xue H，Chen S，Yang Q. Structural support vector machine[C]// Proceedings of the fifth International Symposium on Neural Networks，2008. Berlin，Heidelberg：Springer，2008：501-511

[12] Shalev-Shwartz S，Singer Y，Srebro N. Pegasos：Primal Estimated sub-GrAdient SOlver for SVM[C]// Proceedings of the 24th International Conference on Machine Learning，2007. New York：ACM，2007：807-814