基于噪声自检测的并行 AdaBoost 算法

徐 坚 陈优广
(华东师范大学 上海 200062)

摘 要 P-AdaBoost 通过改良使传统 AdaBoost 算法的核心步骤可以被并行执行,极大提高了算法的执行效率。然而 P-AdaBoost 没有考虑到噪声样本对训练结果造成的负面影响。通过分析 P-AdaBoost 算法,修改原算法中初始权重分布,并提出一种噪声检测算法,改良 P-AdaBoost 算法在带有噪声数据集上的性能。实验结果表明,改进后的算法与原 P-AdaBoost 算法相比,在带有噪声的数据集上提高了将近 5 个百分点,在无噪声的数据集上也有一定提高。由此证明,提出的算法是一种更健壮的算法,在大部分数据集上均取得更高的分类准确率。

关键词 daBoost 数据挖掘 并行化 噪声自检测 分类

中图分类号 TP3 **文献标识码** A DOI:10.3969/j.issn.1000-386x.2018.01.050

PARALLEL ADABOOST ALGORITHM BASED ON NOISE DETECTION

Xu Jian Chen Youguang
(*East China Normal University*, *Shanghai 20006*, *China*)

Abstract P-AdaBoost improved the traditional AdaBoost algorithm flow, the core steps of AdaBoost algorithm can be implemented in parallel to improve the efficiency of the algorithm. However P-AdaBoost does not take into account the negative impact of noise samples on training results. The original P-AdaBoost algorithm is analyzed to modify the initial weight distribution in the original algorithm, and a noise detection algorithm is proposed to improve the performance of the P-AdaBoost algorithm with the noise data set. The experimental results show that the improved algorithm has improved by almost 5 percentage points compared with the original P-AdaBoost algorithm in the data set with noise and no-noise data set. It is proved that the proposed algorithm is a more robust algorithm, and achieves higher classification accuracy in most data sets.

Keywords AdaBoost Data mining Parallelization Noise detection Classify

0 引 言

数据分类问题作为数据挖掘领域中一个非常重要的研究方向,主要由两个阶段构成:学习和分类。学习阶段使用已知类别标记的数据集作为学习算法的输入,构筑分类器,而分类阶段则使用上一阶段得到的分类器预测未知标记数据的类别[1]。用于构筑分类器的方法有很多,如 SVM、决策树和朴素贝叶斯模型等。在这些算法不断成熟完备的过程中,出现了诸如装袋(Bagging)、提升(Boosting)和随机森林等提高分类准确率的技术,这些被称为集成学习技术[2]。集成学习的原理为通过一定规则将一系列弱分类器集成提升为强分类器,再使用强分类器预测未知样本的类别标记。集成算法中,最具代表性的就是 AdaBoost 算法,该算法基于 PAC 框架提出,是集成算法中应用最为广泛的算法[3]。AdaBoost 算法在诸如人脸识别,文本分类等诸多领域表现出众,算法通过反复迭代,可以将弱分类器的集合提升为强分类器,并且使训练误差以指数速度下降,训练集上的错误率趋近 0[4]。但是 AdaBoost 算法同时也有一些缺点,Adaboost 的每次迭代都依赖上一次迭代的结果,这就造成了算法需要严格按照顺序进行,随着迭代次数的升高,算法的时间成本也会相应的增大。基于这个缺点,Stefano Merler 在 2006 年时提出了 P-AdaBoost 算法。利用 AdaBoost 在经过一定迭代次数的训练之后,并行地求出剩余弱分类器对应

收稿日期:2017-04-16。徐坚,硕士生,主研领域:图像处理,人工智能。陈优广,副教授。

权重的训练误差,并证明了在经过足够次数的顺序迭代之后,并行 AdaBoost 的分类准确率跟传统 AdaBoost 相比基本相同甚至更高。然而他们没有考虑到数据集中存在的噪声会对结果造成负面的影响。AdaBoost 本身是一个噪声敏感的算法,这是由于该算法在每次迭代时,都会将上次训练中分错的样本的权重加大,让本次训练的弱分类器更加关注这个被分错的样本。如果该错分样本是一个噪声,那么这个本来是噪声的样本的权重会在后续迭代过程中不断加大,导致后续弱分类器的分类准确率下降,最终导致整个模型的准确率随之下降[5]。

同时,在 P-AdaBoost 算法中,噪声比例的上升会导致算法构筑分类模型需要顺序迭代的次数变多,时间成本变高,得到的最终模型的分类准确度也会降低[6]。本文为了提高并行 AdaBoost 算法的分类准确率和健壮性,选择基于 P-AdaBoost 算法进行改进,在核心算法流程上保留了其构建模型时间较短的优势。同时通过修改权重分布的初始值,并利用噪声自检测方法减少数据集中的噪声对训练结果的影响,提出了一种名为 ND-PAdaBoost (nosie-detection PAdaBoost) 的算法,进一步提升算法训练结果的准确度。

1　AdaBoost 算法概述

Adaboost 的基本思想是通过不断迭代,利用已知类别标记的训练集训练出一系列弱分类器,然后将这些弱分类器线性结合在一起,得到一个分类准确率更高的强分类器。基分类器的训练算法种类很多,可以使用决策树、朴素贝叶斯等,还有学者提出也能使用神经网络作为算法的基分类器[7]。

AdaBoost 算法流程如下:

输入:

1) 含有 N 个已知样本的训练集:

$$S = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\} \tag{1}$$

2) 任意作为基分类器的分类算法。

输出:集成的最终模型 $G(x)$。

步骤 1　初始化训练集每个样本的训练权重,均匀分布,每个样本赋值相同的权重。表示为:

$$\begin{aligned} D_1 &= (w_{11}, w_{12}, \cdots, w_{1i}, \cdots, w_{1N}) \\ w_{1i} &= \frac{1}{N} \quad i = 1, 2, \cdots, N \end{aligned} \tag{2}$$

步骤 2　以 T 表示算法迭代的轮数,t 表示是第 t 次迭代,D_t 表示第 t 次迭代之后的样本权重分布,对于每一轮迭代有如下操作:

(1) 使用设置好权重 D_t 的训练集和选择的分类器算法进行训练,得到一个基分类器 $h_t(x) \in \{-1, +1\}$。

(2) 计算上一步中得到的分类器在训练集上分类的误差率,记为 e_t:

$$e_t = P(h_t(x) \neq y_i) = \sum_{i=1}^{N} w_{ti} I(h_t(x_i) \neq y_i) \tag{3}$$

其中: $I(x)$ 为指示函数,在 x 为真时取值为 1,反之则为 0。

(3) 检查误差率 e_t,如果 $e_t > 0.5$,迭代轮次减 1,将样本的权重初始化为 $\frac{1}{N}$,重新训练基分类器;如果 $e_t < 0.5$,继续进行下一步。

(4) 利用得到的该分类器的误差率,计算该分类器对应的权重为:

$$\alpha_t = \frac{1}{2} \ln\left(\frac{1 - e_t}{e_t}\right) \tag{4}$$

该权重表示当前分类器在最终模型中的比重,也是最终模型中该分类器对应的系数。

(5) 更新训练数据集的样本权重,$D_{t+1} = (w_{t+1,1}, w_{t+1,2}, \cdots, w_{t+1,i}, \cdots, w_{t+1,N})$,有:

$$w_{t+1,i} = w_{t,i} \times \frac{e^{-\alpha_t y_i h_t(x_i)}}{Z_t} \tag{5}$$

$$Z_t = \sum_{i=1}^{N} w_{t,i} \times e^{-\alpha_t y_i h_t(x_i)} \tag{6}$$

其中: Z_t 作为归一化因子,用以确保每一轮的样本权重和为 1。

步骤 3　将经过迭代后得到的一系列基分类器及其相应权重线性组合得到最终分类器,表示为:

$$G(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t \times h_t(x)\right) \tag{7}$$

2　P-AdaBoost 算法概述

P-AdaBoost 算法利用 AdaBoost 中权重分布的特性,通过一定次数的顺序迭代,用一种概率分布拟合样本权重的分布函数,使得后续训练得到的基分类器不用再通过上一次训练的结果更新计算本轮的权重,让后续的训练过程能够并行执行,优化了算法的时间成本,算法的建模效率得到明显提升。

2.1　权重动态分布

一权重的动态分布包含着许多关键的信息,这些信息在 AdaBoost 算法构建模型的时候起到了至关重要的作用[8]。AdaBoost 训练基分类器时可以将样本简单地分为 2 类:易分类样本和难分类的样本。易分类样本在训练分类器的时候不容易被分错,根据 AdaBoost 算法原理,这些样本的权重会逐轮次降低,对新

分类器训练起到的影响会越来越小。而难分类样本的权重并不会向一个固定值收敛,随着迭代次数的增加可能会发生随机波动。

一些研究人员的工作证明了难分类样本的权重分布遵从以下两个事实:(1) 对于任意一个难分类点,在基分类器数趋于无穷的情况下,该点的权重分布收敛于一个可确定的稳定分布;(2) 这一分布能够用参数选择适当的伽马分布来表示[9],这使得算法能够并行计算出每轮迭代的样本权重。

2.2 P-AdaBoost 算法流程

输入:

1) 含有 N 个已知样本的训练集 $S = \{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$;

2) 任意作为基分类器的分类算法。

输出:集成的最终模型 $G(x)$。

步骤 1 初始化训练集每个样本的训练权重,表示为 $D_1 = (w_{11},w_{12},\cdots,w_{1i},\cdots,w_{1N}),w_{1i} = \frac{1}{N},i = 1,2,\cdots,N$。

步骤 2 顺序迭代 S 轮的 AdaBoost 算法,保存每一轮的到的权重结果 $w_i(s),s = 1,2,\cdots,S$。

步骤 3 根据 $w_i(s)$ 估计该权重的分布函数 r_i^*。

步骤 4 对 $S = S+1,S+2,\cdots,T$ 做如下操作(该循环能够并行执行)。

(1) 对于第 i 次迭代,通过对 r_i^* 采样随机生成的的权重值 $w_i^*(s)$。

(2) 使用 $w_i^*(s)$ 权重训练得到新的分类器 h_s。

(3) 计算该模型对应的误差率 e_s。

如果 $e_s > 0.5$,将所有权重初始化为 $\frac{1}{N}$,重新训练分类器 h_s。

如果 $e_s < 0.5$,进行下一步。

(4) 根据误差率计算该分类器在总分类器函数中的权重系数 $\alpha_t = \frac{1}{2}\ln\left(\frac{1-e_t}{e_t}\right)$。

步骤 5 输出最终模型:

$$G(x) = sign\left(\sum_{t=1}^{T}\alpha_t \times h_t(x)\right)$$

2.3 P-AdaBoost 算法分析

P-AdaBoost 算法的核心为采用伽马分布取代了原来的权重更新函数,为了求出这个伽马分布 r_i^*,算法需要按照传统 AdaBoost 的方式顺序迭代足够的轮次,通过记录得到的 $w_i(s)$,求出伽马分布为:

$$\gamma(x) = \frac{x^{\alpha-1}e^{\frac{-x}{\theta}}}{\Gamma(\alpha)\theta^{\alpha}} \tag{8}$$

式中:α 和 θ 的值是通过均值 μ 和方差 σ^2 求解得到的,有如下关系:

$$\mu = \alpha\theta \tag{9}$$

$$\sigma^2 = \alpha\theta^2 \tag{10}$$

然而,P-AdaBoost 算法的缺点也很明显,由于在算法的并行建模阶段生成的 $w_i^*(s)$ 都是随机采样的,造成该算法对噪声敏感。其原因在于,像 AdaBoost 这类算法可以视作基于训练集对损失函数进行的梯度下降过程[10],每一个子模型加入到最终模型时可以视作为一次线性的偏移。AdaBoost 算法就是沿着损失函数下降最快的方向进行偏移的。然而由于 P-AdaBoost 算法在权重选择上的随机采样特点,导致其偏移的方向可能会与传统的 AdaBoost 算法偏移方向不同。因此 P-AdaBoost算法需要足够的顺序迭代次数保证 r_i^* 分布的准确性,也依赖于一个理想的训练集(没有噪声)。训练集中的噪声样本会降低 r_i^* 分布的准确性,错误的 r_i^* 会导致 P-AdaBoost 在经过一定轮次的迭代之后无法取得跟 AdaBoost 算法相近或者更优的结果,这极大地限制了 P-AdaBoost 的应用。本文通过加入噪声自检测算法,成功解决了这一问题,增强了 P-AdaBoost 算法的健壮性,拓展了该算法的应用场景,使其在含有噪声的数据集中依然能够取得优秀的分类表现。

3 ND-PAdaBoost 算法概述

3.1 数据集不平衡问题

数据集中的数据分布不平衡问题在分类问题中经常出现,其主要表现为某一类的样本数远大于其他类别的样本数。而少数类又恰好是最需要学习的概念,由于少数类数据可能和某一特殊、重要的情形相关,造成了这类数据难以被识别。

诸如 AdaBoost、P-AdaBoost 等标准的学习算法考虑的是一个平衡数据集,当这些算法用到不平衡数据集时,可能会生成一个局部优化的分类模型,导致经常错分少数类样本[11]。因此 AdaBoost 和 P-AdaBoost 在平衡数据框架下分类准确率较高,然而在处理不平衡数据集时,分类准确率可能会下降。造成这种现象的主要原因如下:

(1) 分类准确率这种用以指导学习过程的,衡量全局性能的指标可能会偏向多数类,对少数类不利。

(2) 预测正例样本的分类规则可能非常的特殊,其覆盖率很低,训练过程中适应那些少数类的规则可能被丢弃。

(3) 规模非常小的少数类可能会被错误的当做噪

声数据,同时真正的噪声会降低分类器对少数类的识别性能。

因此本文认为 P-AdaBoost 和 AdaBoost 中权重的初始值 $\frac{1}{N}$ 不太合理。本文通过推理和实验发现,对数据集中的正样本和负样本分别赋不同的初始值 $\frac{1}{2m}$ 和 $\frac{1}{2l}$ 时,对最终模型的分类准确率提升很有帮助,其中 m 和 l 分别为正样本和负样本的个数,1/2 是为了消除随机误差。这是因为 $\frac{1}{N}$ 取值的本意是对 N 个样本赋相同的初始权值,为了保证在区别对待正例和反例的同时不会产生较大的误差,权重取值需要乘上 1/2 这个系数。

3.2　数据集中噪声样本的检测

真实世界中的数据集包含着各种各样的噪声。由于 AdaBoost 算法框架的特性,这些包含在数据集中的噪声会对算法生成的最终模型造成负面影响,导致过拟合的出现。造成这种结果的原因是 AdaBoost 算法在每次迭代之后会将之前分错的样本的权重提高,下一次迭代训练的分类器会着重针对这类权重高的样本学习,以此来最小化损失函数。然而由于大部分的噪声样本都不符合分类器的学习规则,导致噪声样本是很难被分类正确的。传统的 AdaBoost 没有对这种噪声样本进行处理,导致了这些样本随着迭代轮次的增长权重越来越大,后续训练的分类器偏向于这些权重大的噪声样本进行训练,使整个模型容易过拟合。而那些真正需要被正确分类的样本没有得到充分的学习,分类器的分类准确度下降,最终造成集成模型的分类准确度下降[12]。

P-AdaBoost 算法依赖于传统 AdaBoost 算法的框架,也会面临与 AdaBoost 同样的问题——对噪声敏感。此外,由于 P-AdaBoost 算法的核心部分是通过一定轮次迭代的传统 AdaBoost 得到样本权重分布 r_i^*。因此样本集中噪声比例的升高会导致根据记录的 $w_i(s)$ 得到的权重分布 r_i^* 并不能可靠的取代原本的权重更新函数,进而导致 P-AdaBoost 的分类准确率无法在基分类器数足够多时收敛,严重限制了 P-AdaBoost 算法的应用场景。同时,考虑到后续并行训练弱分类器的过程中每次都重新随机生成了样本权重,不能利用修改噪声样本权重的方式控制噪声对训练过程的影响,本文采用了噪声移除的方式。在进入并行步骤之前做一次噪声检测,将被算法认定为噪声的样本从数据集中移除。采用更新后的数据集作为输入进行后续并行训练弱分类器的流程。

为了找出样本集中包含的噪声样本,本文考虑了这样的一种思路,因为每一个样本都是样本空间中的一个点。对于一个样本点 (x_i,y_i),已知一个分类模型 $h_t(x)$,如果该点在分类模型上取值 y_i 与它邻近的点都不同,那么就有理由相信该点其实是一个噪声点。这是因为在分类的过程中,相同分类的点针对于一个已经训练出来的分类模型 $h_t(x)$ 的取值总是相近的。这标志了它们属于同一种分类并且这些点往往都集中分布在近邻的位置。如果这时有一个邻近点(其欧氏距离在一定范围内)的取值与其相邻的点在 $h_t(x)$ 上的取值都不同,这就说明了这个点即不能划归到另外一个分类。在欧氏距离上该点跟这些已分类点邻近,又不能将该点划为本分类,因为分类器的取值不同,所以该点就是一个噪声点。基于这一思想,本文提出了一种基于 k 邻近算法的噪声检测算法,并采用模拟随机噪声的方式进行验证,由于是随机噪声,所以无需在算法中针对噪声的分布做处理,算法流程如下:

输入:样本集 $S=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$,在 S 上训练得到的分类器 $h_t(x)$,k 邻近算法的系数 k。

输出:噪声样本集 NS 和非噪声样本集 NNS。

步骤 1　对于每一个样本 (x_i,y_i) 循环进行如下操作:

利用 k 临近算法找到该点最近的 k 个相邻点(以欧氏距离为基准),记为 $\{(x_{ij},y_{ij})\}_{j=1}^{k}$;

计算 (x_i,y_i) 点可能为一个噪声点的概率为:

$$\mu(x_i,y_i)=\left(\frac{1}{k}\right)\sum_{j=1}^{k}I(h_t(x_{ij})\neq y_{ij}) \tag{11}$$

其中:$I(x)$ 为指示函数,在 x 为真时取值为 1,反之则为 0。

步骤 2　计算此概率在整个样本集上的平均为:

$$\bar{\mu}=\left(\frac{1}{N}\right)\sum_{i=1}^{N}\mu(x_i,y_i) \tag{12}$$

步骤 3　对于 S 上的每一个样本 (x_i,y_i) 做如下操作:

(1) 如果 $\mu(x_i,y_i)>\bar{\mu}$,该样本被标记为噪声,加入噪声集 NS:$NS=NS\cup\{(x_i,y_i)\}$;

(2) 如果 $\mu(x_i,y_i)<\bar{\mu}$,该样本被标记为非噪声,加入噪声集 NNS:$NNS=NNS\cup\{(x_i,y_i)\}$。

3.3　算法流程

考虑到迭代部分和并行部分对噪声处理的需求不同,本文在顺序迭代部分只将噪声的权重置为 0,在进入并行流程之前用当前的分类器对数据集进行一次噪声监测,只保留非噪声样本为新的训练数据集。

详细的算法流程如下:

输入:

1) 含有 N 个已知样本的训练集 $S=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$,其中 x_i 为样本矢量,$y_i\in\{-1,+1\}$;

2) 任意作为基分类器的分类算法。

输出:集成的最终模型 $G(x)$。

步骤 1 初始化训练集每个样本的训练权重,表示为 $D_1=(w_{11},w_{12},\cdots,w_{1i},\cdots,w_{1N})$,对于正样本:$w_{1i}=\frac{1}{2m}$,负样本 $w_{1i}=\frac{1}{2l}$,$i=1,2,\cdots,N$,m = 正样本个数,l = 负样本个数;

步骤 2 顺序迭代 S 轮的 AdaBoost 算法,在得到本轮训练生成的分类器 $h_t(x)$ 时,使用分类器对当前数据集运行噪声检测算法,将标记为噪声样本的权重设置为 0,非噪声样本的权重按照传统算法进行更新。保存每一轮的到的权重结果 $w_i(s)$,$s=1,2,\cdots,S$;

步骤 3 使用当前生成的集成分类器:

$$G'(x)=\mathrm{sign}\left(\sum_{s=1}^{S}\alpha_s\times h_s(x)\right)$$

对数据集运行噪声检测算法,得到 NS 和 NNS,使用 NNS 作为新的数据集 S';

步骤 4 根据 $w_i(s)$ 估计该权重的分布函数 r_i^*;

步骤 5 对 $S=S+1,S+2,\cdots,T$ 做如下操作(该循环能够并行执行):

(1) 对于第 i 次迭代,通过对 r_i^* 采样随机生成的的权重值 $w_i^*(s)$;

(2) 使用 $w_i^*(s)$ 权重在新的数据集 S' 上训练得到新的分类器 h_s;

(3) 计算该模型对应的误差率 e_s。

如果 $e_s>0.5$,将所有权重初始化为正样本 $w_{1i}=\frac{1}{2m}$,负样本 $w_{1i}=\frac{1}{2l}$,重新训练分类器 h_s;

如果 $e_s<0.5$,进行下一步;

(4) 根据误差率计算该分类器在总分类器函数中的权重系数 $\alpha_t=\frac{1}{2}\ln\left(\frac{1-e_t}{e_t}\right)$;

步骤 6 输出最终模型:

$$G(x)=\mathrm{sign}\left(\sum_{t=1}^{T}\alpha_t\times h_t(x)\right)$$

4 实 验

4.1 数据集

本文采用 UCI(加州大学欧文分校)提供的机器学习公共数据集(见表 1),用以验证本文的算法相比于 P-adaBoost、传统的 AdaBoost 和 Bagging 算法在相同数据集下分类结果的准确率更高。

表 1 数据集

数据集名称	属性数	样本数
German	20	1 000
Breast c.	9	277
Heart	13	270
Diabetes	8	768
Thyroid	5	215
Titanic	3	2 201
Banana	2	5 300

4.2 实验方法

本文实验基于 Weka 平台,使用 Java 语言实现了 ND-PAdaBoost 算法。由于受实验条件所限,只模拟了 ND-PAdaBoost 算法并行运行的情况,由于模拟并不改变算法更新权重值的逻辑,所以其结果的分类准确率与算法真正并行运行的分类准确率一致,不影响对 ND-PAdaBoost 算法的评估和比较。实验对比了本文提出的 ND-PAdaBoost 算法和 P-AdaBoost、传统 AdaBoost、Bagging 这四种算法在无噪声情况下的分类准确率。由于这四种算法流程上存在并行训练和连续迭代训练的区别,为了能充分验证本文提出算法 ND-PAdaBoost 的效果,该实验规定除 Bagging 算法外,其他含有连续迭代过程的算法的迭代轮次为 100 次。保证前三种算法的时间成本基本相同,并且本文通过反复实验证明了 100 次的顺序迭代足够使 ND-PAdaBoost 和 P-AdaBoost 算法估计出可信的样本权重分布。同时规定 ND-PAdaBoost、P-AdaBoost 和 Baggaging 这三种并行算法训练的总基分类器数目为 500 个,k 邻近算法的系数设置为 5%。实验记录各算法在无噪声下的分类准确率。

此外,为了说明本文改进权重初始值对算法分类准确率的提高产生了正面影响。本文在上述所有数据集上分别使用不同的初始值下的 ND-PAdaBoost 算法训练分类模型,NDP-AdaBoost 的其余条件保持不变。通过比较最终分类模型的准确率来证明修改后的权重初始值有助于提高最终模型的分类准确率。

4.3 实验结果

实验结果如表 2 所示,其中加粗部分是每组数据集中正确率最高的,下划线部分是每组数据集中分类正确率最低的。通过该实验结果表明,ND-PAdaBoost 算法即使在无噪声的数据集上的分类准确率相比于其他三种算法也有所提升。

表 2　无噪声下各算法的分类准确率(100 次迭代)

数据集	ND-PAda-Boost	P-Ada-Boost	Ada-Boost	Bag-ging
German	**81.537 8**	79.021 3	78.631 2	79.452 4
Breast c.	**79.097 2**	75.552 3	75.230 8	77.253 2
Heart	**86.262 3**	83.523 2	82.252 1	84.456 2
Diabetes	**81.568 2**	78.170 5	78.349 0	77.897 6
Thyroid	**96.235 1**	95.845 2	95.422 5	94.246 4
Titanic	**82.642 2**	78.255 3	77.753 1	79.025 1
Banana	**93.598 3**	88.286 3	85.053 3	88.203 5

取 Titanic、Diabetes 数据集为例,比较前三种算法在不同噪声比例下的分类准确率。结果如表 3 所示。

表 3　不同噪声比例下各算法分类准确率

数据集名称	噪声	ND-PAdaBoost	P-AdaBoost	AdaBoost
Titanic	5%	80.642 2	77.255 3	76.753 1
	10%	78.921 2	72.642 1	72.371 2
	15%	76.242 4	67.253 1	65.262 1
	20%	74.982 1	61.529 6	60.124 2
Diabetes	5%	79.568 2	75.170 5	74.349 0
	10%	78.008 2	70.935 1	69.836 2
	15%	75.920 4	65.920 4	63.124 2
	20%	73.525 7	61.242 4	59.262 3

根据每组在不同噪声比例下的分类准确率绘制如图 1 和图 2 所示。

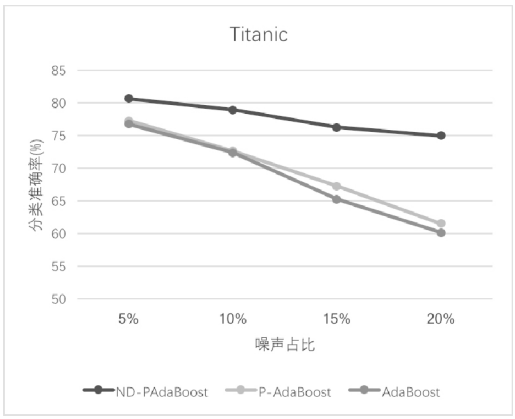

图 1　Titanic 数据集折线图

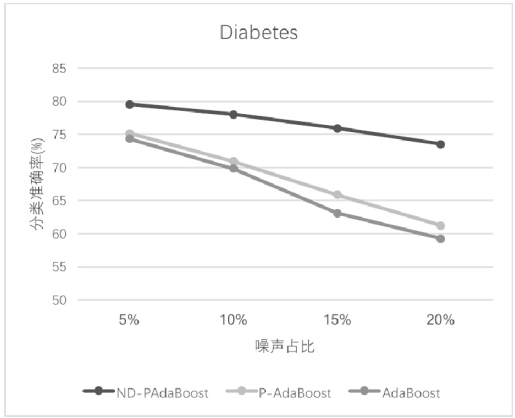

图 2　Diabetes 数据集折线图

由图 1、图 2 可以看出,ND-PAdaBoost 算法相比于 P-AdaBoost 和传统的 AdaBoost 对噪声有更好的兼容性,能够在含有噪声的数据集上取得更稳定的表现。随着噪声比例的增加,ND-PAdaBoost 算法的分类准确率与 P-AdaBoost 和 AdaBoost 差距明显,一直维持在较高的水平。

将权重初始值分别取值为 $\frac{1}{N}$,$\frac{1}{2m}$(正例)和$\frac{1}{2l}$(反例),使用 ND-PAdaBoost 训练分类器,分类准确率如表 4 所示。

表 4　不同权重初始值下 ND-PAdaBoost 算法分类准确率

数据集	$\frac{1}{2m},\frac{1}{2l}$	$\frac{1}{N}$
German	**81.537 8**	79.340 1
Breast c.	**79.097 2**	75.038 2
Heart	**86.262 3**	83.371 5
Diabetes	**81.568 2**	77.963 2
Thyroid	**96.235 1**	95.362 5
Titanic	**82.642 2**	78.742 1
Banana	**93.598 3**	88.123 7

由表 4 可以看出,在大部分数据集上,$\frac{1}{2m}$(正例)和$\frac{1}{2l}$(反例)的初始值方案比起全部赋值$\frac{1}{N}$的方案确实提高了一定的分类准确率。这说明本文采取的修改初始值的改进方法能够在一定程度上提高最终结果的分类准确率。

5　结　语

本文从 AdaBoost 算法的框架出发,基于 AdaBoost 的并行改进算法 P-AdaBoost,针对 P-AdaBoost 对噪声敏感,含有噪声的数据集会严重影响 P-AdaBoost 分类准确度的问题,提出了一个噪声自检测方法,并修改了原始样本权重分布的初始值。本文使用了 UCI 数据集对提出的改进方法进行了多番验证。实验结果表明,本文提出的算法在输入数据集含噪声的场景下具有更好的分类准确率。然而,本文提出的模型只准对二分类问题有效,如果需要推广到多分类问题上,还需要对模型进行修改。本算法中所需顺序迭代次数的取值是通过反复实验测定的,关于这个取值的最优数值还需要进一步的研究和探讨。

参　考　文　献

[1] 周志华. 机器学习 [M]. 北京:清华大学出版社,2016: 121-145.

[2] Breiman L. Bagging predictors [J]. Machine Learning,1996,24(2):123-140.
[3] Breiman L. Random Forests [J]. Machine Learning,1996,24(2):123-140.
[4] Nie Q,Jin L,Fei S. Probability estimation for multi-class classification using AdaBoost [J]. Pattern Recognition,2014,47(12):3931-3940.
[5] 曹莹,苗启广,刘家辰,等. AdaBoost 算法研究进展与展望 [J]. 自动化学报,2013,39(6):745-758.
[6] Merler S,Caprile B,Furlanello C. Parallelizing AdaBoost by weights dynamics [J]. Computational Statistics & Data Analysis,2007,51(5):2487-2498.
[7] 李翔,朱全银. Adaboost 算法改进 BP 神经网络预测研究 [J]. 计算机工程与科学,2013,35(8):96-102.
[8] Caprile B,Furlanello C,Merler S. Highlighting Hard Patterns via AdaBoost Weights Evolution [J]. Lecture Notes in Computer Science,2002,2364:72-80.
[9] Collins M. Logistic regression,Adaboost and bregman distances [J]. Machine Learning,2002,48(1):253-285.
[10] Htike K K. Efficient determination of the number of weak learners in AdaBoost [J]. Journal of Experimental & Theoretical Artificial Intelligence,2016,29(5):1-16.
[11] Lee W,Jun C,Lee J. Instance categorization by support vector machines to adjust weights in AdaBoost for imbalanced data classification [J]. Information Sciences,2017,381:92-103.
[12] Barrow D K,Crone S F. A comparison of AdaBoost algorithms for time series forecast combination [J]. International Journal of Forecasting,2016,32(4):1103-1119.

(上接第 122 页)

模型 ARIMA(3,0,1) × (1,1,1)$_{24}$能很好地拟合测试数据。

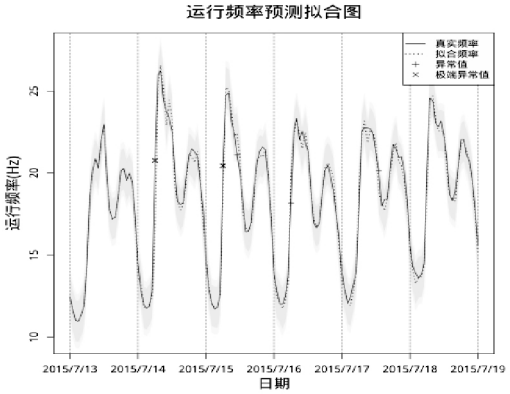

图 7 运行频率预测拟合图

4 结 语

通过对所采集的数据进行数据预处理,对供水管网的实时数据及历史数据的分析,建立季节性 ARIMA 模型并进行预测。实验结果表明,得到的季节性 ARIMA 模型能够有效地对小区的供水状况进行预测。该方法具有较好实际应用价值,可以为降低小区供水的运维成本提供参考。在后续工作中,可以根据预测供水情况,通过远程智能控制等手段,实现更高效的远程控制,以进一步提升系统的整体效率,达到节能的目的。

参 考 文 献

[1] 韩晓峰,丁莉芬. 浅析远程监控系统在城市二次供水管理中的应用 [J]. 给水排水,2014(4):120-123.
[2] 廖曙江,刑佳佳,陈睿迪,等. 基于物联网技术的远程建筑消防水压实时监控系统 [J]. 自动化与仪器仪表,2012(5):16-17,20.
[3] 付刚,朱晨光,刘彦华. 基于远程控制技术的二次供水管理系统的探索与实践 [J]. 中国给水排水,2013,29(12):14-17.
[4] 练庭宏,刘秋娟,王景成. 基于 ARIMA 时序辨识的需水量预测 [J]. 控制工程,2008(S1):166-168.
[5] Firat M,Turan M E,Yurdusev M A. Comparative analysis of neural network techniques for predicting water consumption time series [J]. Journal of Hydrology,2010,384(1):46-51.
[6] 高金良,姚芳,叶健. 结合图论的供水管网 PMA 分区方法 [J]. 哈尔滨工业大学学报,2016,48(8):67-72.
[7] 程伟平,赵丹丹,许刚,等. 供水管网爆管水力学模型与爆管定位 [J]. 浙江大学学报(工学版),2013,47(6):1057-1062.
[8] Wu Yipeng,Liu Shuming,Wu Xue,et al. Burst detection in district metering areas using a data driven clustering algorithm [J]. Water Research,2016,100:28-37.
[9] Hutton C,Kapelan Z. Real-time Burst Detection in Water Distribution Systems Using a Bayesian Demand Forecasting Methodology [J]. Procedia Engineering,2015,119(1):13-18.
[10] 文玉梅,张雪园,文静,等. 依据声信号频率分布和复杂度的供水管道泄漏辨识 [J]. 仪器仪表学报,2014,35(6):1223-1229.
[11] 邹柏贤,刘强. 基于 ARMA 模型的网络流量预测 [J]. 计算机研究与发展,2002,39(12):1645-1652.
[12] 韩超,宋苏,王成红. 基于 ARIMA 模型的短时交通流实时自适应预测 [J]. 系统仿真学报,2004,16(7):1530-1532,1535.
[13] 彭志行,鲍昌俊,赵杨,等. ARIMA 乘积季节模型及其在传染病发病预测中的应用 [J]. 数理统计与管理,2008,27(2):362-368.
[14] 白云. 时间序列特性驱动的供水量预测方法研究及应用 [D]. 重庆大学,2014.