软件学报 ISSN 1000-9825, CODEN RUXUEW
*Journal of Software*,2011,22(9):2059−2074 [doi: 10.3724/SP.J.1001.2011.03827]

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

# 支持在线学习的增量式极端随机森林分类器*

王爱平+, 万国伟, 程志全, 李思昆

(国防科学技术大学 计算机学院,湖南 长沙 410073)

# Incremental Learning Extremely Random Forest Classifier for Online Learning*

WANG Ai-Ping+, WAN Guo-Wei, CHENG Zhi-Quan, LI Si-Kun

(College of Computer, National University of Defense Technology, Changsha 410073, China)

+ Corresponding author: E-mail: ipwang@nudt.edu.cn

**Wang AP, Wan GW, Cheng ZQ, Li SK. Incremental learning extremely random forest classifier for online learning. *Journal of Software*, 2011,22(9):2059−2074.** http://www.jos.org.cn/1000-9825/3827.htm

**Abstract**: This paper proposes an incremental extremely random forest (IERF) algorithm, dealing with online learning classification with streaming data, especially with small streaming data. In this method, newly arrived examples are stored at the leaf nodes and used to determine when to split the leaf nodes combined with Gini index, so the trees can be expanded efficiently and fast with a few examples. The proposed online IERF algorithm gives more competitive or even better performance, than the offline extremely random forest (ERF) method, based on the UCI data experiment. On the moderate training datasets, the IERF algorithm beats the decision tree reconstruction algorithm and other incremental learning algorithms on the performance. Finally, the IERF algorithm is used to solve online video object tracking (multi-object tracking also included) problems, and the results on the challenging video sequences demonstrate its effectiveness and robustness.
**Key words**: online learning; incremental learning; extremely random forest classifier

**摘 要**: 提出了一种增量式极端随机森林分类器(incremental extremely random forest,简称 IERF),用于处理数据流,特别是小样本数据流的在线学习问题.IERF 算法中新到达的样本将被存储到相应的叶节点,并通过 Gini 系数来确定是否对当前叶节点进行分裂扩展,在给定有限数量,甚至是少量样本的情况下,IERF 算法能够快速高效地完成分类器的增量构造.UCI 数据集的实验证明,提出的 IERF 算法具有与离线批量学习的极端随机森林(extremely random forest,简称 ERF)算法相当甚至更优的性能,在适度规模的样本集上,性能优于贪婪决策树重构算法和其他几种主要的增量学习算法.最后,提出的 IERF 算法被应用于解决视频在线跟踪(包含多目标跟踪)问题,基于多个真实视频数据的实验充分验证了算法的有效性和稳定性.
**关键词**: 在线学习;增量学习;极端随机森林分类器
**中图法分类号**: TP181 **文献标识码**: A

传统的机器学习往往采用批量学习的方法,即所有的训练样本一次性学习完毕后,学习过程不再继续.但在实际应用中,训练样本空间的全部样本并不能一次全部得到,而往往是随着时间顺序得到.考虑到训练和预测的

* 基金项目: 国家自然科学基金(90707003, 60970094)
收稿时间: 2009-08-22; 修改时间: 2009-10-23; jos 在线出版时间: 2010-07-01

时空开销需求,能够在已有训练结果的基础上继续学习新样本,不断增强模型本身的识别能力,并且减少重复学习的时空开销的增量学习方法,得到了广泛的关注.增量学习目前没有严格的定义,但其主要特征包含两点[1,2]:能够将新样本加入到已有的知识系统中;能够使一个基本的知识系统逐步演化为更加复杂的系统.增量学习包含很多方法,按照分类器的种类划分,有基于支持向量机的增量学习算法[3−5]、基于最近邻方法的增量学习算法[6],基于 Bagging/Boosting 的增量学习算法[7]、基于决策树的增量学习算法[8−16],以及基于贝叶斯网络的增量学习算法[17]和基于 RBF 网络的增量学习算法[18]等.

基于决策树的方法目前已被广泛用于增量学习,一是由于决策树本身简单、快速,决策树模型易于理解,通过决策树的工作过程可以直观理解问题的求解过程.二是由于决策树能够对广泛的问题给出准确的解[13].决策树通常将多维数据迭代的划分为若干更小维数的数据进行处理,这种分治策略(divide-and-conquer)适用于诸多问题的求解.三是由于决策树的构造效率很高.对于 $n$ 个样本和 $m$ 个属性,文献[19]证明了忽略数值属性等复杂因素,决策数的平均构造开销是 $O(m\cdot n\cdot \log n)$.

按照树模型来划分,当前增量决策树算法研究可分为两个主要方向:第 1 类是采用贪婪算法重构决策树,例如 ID5R[8],ITI[9]及其改进方法[10].这类方法将新样本加入到决策树以后,需要重新确定分类路径上每个决策节点的决策属性,以确保该决策属性是最适合该节点的.若选择的最优属性不在当前决策节点上,则要将其从叶节点提升(pull up)上来,并迭代地为其子树上所有决策节点确定最优属性.由于每新加入一个样本都需要对树进行变换重构,因此当问题规模达到一定程度后,这类方法的计算时间开销非常大.第 2 类方法是通过维护一些必要的统计量来完成决策树的增量构造,例如 VFDT[11,12],IADEM[16],Streaming Random Forest[15],Improving Hoeffding Trees[13]以及相应的改进方法[14].这些方法采用概率统计量,例如 Hoeffding 边界[11](也被称为 Chernoff 边界),主要用于确定当前叶节点是否需要分裂成为决策节点.新样本到达叶节点后不被存储,而是更新该叶节点上的统计信息,当有足够多的样本到达该节点后,根据统计信息可以判定是否需要将该叶节点分裂为决策节点.当叶节点分裂为决策节点后,统计信息会传递到后继子节点,决策节点仅保留最佳决策属性.这类方法无须存储训练样本,因而可耗费低时空开销来处理海量数据,但是只有当接收足够多的样本信息后,才能有充分的统计证据确定是否进行节点分裂.对于采用 Hoeffding 边界作为统计量的方法来说,针对二值分类问题,平均需要 $5\times10^6$ 个样本来构造一棵决策树,而针对多类分类问题例如 $C$-类分类问题,构造一棵决策树通常需要 $(5\times10^6)/2\times C$ 个样本[15].虽然具有很好的增量学习性能,但第 2 类方法不适合小样本数据流的应用.

在实际应用中,小样本数据流的问题很普遍,例如在线视频跟踪问题.当前视频跟踪常作为分类问题进行解决,在线学习的分类器对于目标物体外观的改变和复杂的环境具有很好的适应性.增量学习分类器可以很自然地应用于在线学习的应用.Feng[20]采用增量学习SVM方法[2],结合半指导学习理论[21],利用视频图像中大量的无标记样本完成分类器的在线学习,得到了稳定可靠的跟踪结果.但文中的算法不能处理多目标识别跟踪,并且目标物体大尺度和大方向的变化情况也不能进行有效地跟踪.

为此,本文提出了一种增量式极端随机森林(incremental extremely random forest,简称 IERF)分类器,能够有效解决小样本数据流的在线学习问题,并在视频、在线跟踪(包含多目标跟踪)等应用上的得到了实验验证.

本文研究做出贡献如下:

1) 设计了一种支持在线学习的IERF分类器,在给定有限数量的样本特别是小样本的情况下,本文提出的算法能够对分类器进行快速有效的扩展,并能达到与离线批量学习的极端随机森林[22](extremely random forest,简称ERF)分类器相当甚至更优的性能,在时间开销和分类正确率等方面优于当前贪婪决策树重构算法和其他几种主要的增量学习算法.

2) 将本文提出的IERF算法应用于解决小样本数据流的一个应用——视频在线跟踪问题,本文提出的基于 IERF 分类器的在线视频跟踪算法,能够在目标外形发生大尺度变化以及复杂背景等情况下,实现对目标稳定可靠的跟踪,并且具有多目标跟踪的能力.

## 1 随机森林分类器与极端随机森林分类器

随机森林是一种包含多棵决策树的分类器,其中每棵决策树的构造和分类测试均相互独立.训练过程中,每棵决策树对原始的训练数据集进行采样替换,构造新的训练数据集(bootstrap方法);决策树中每个决策节点上的分裂测试均从一个随机测试集合中产生:根据某种量化评价标准,例如信息熵等,从随机测试集合中选择一个最佳测试作为决策点的分裂测试.随机森林中的每棵决策树均不需要进行剪枝.假设随机森林分类器记为 $F(\vec{x})$,其中第$i$棵决策树记为 $f_i(\vec{x}), F(\vec{x})=\{f_i(\vec{x}), i\in[0,N]\}$, $N$为随机森林中所有决策树的数量, $\vec{x}$ 为输入的待分类的样本, $c(\vec{x})\in C=\{1,2,\ldots,M\}$ 为 $\vec{x}$ 的类别标记值, $f_i(\vec{x})$ 的输出为$C$中的某个值,作为单棵决策树对样本 $\vec{x}$ 类别的估计值, $F(\vec{x})$ 的输出为 $\{f_i(\vec{x}), i\in[0,N]\}$ 估计值的众数.

相对于单棵决策树而言,随机森林可以避免过拟合问题,分类精度高,稳定性好;随机森林保留了多值分类的特性,适合处理多值分类问题;相比boosting和其他集成方法,随机森林方法对于数据噪声更稳定[23].

极端随机森林同样是一种多棵决策树集成(ensemble)的分类器.与随机森林分类器相比,主要有两点不同:一是不采用bootstrap采样替换策略,而是直接采用原始的训练样本,目的在于减少偏差(bias)[22];二是在每棵决策树的决策节点上,分裂测试的阈值是随机选择的.假设决策节点上的分裂测试表达为 $split(\vec{x})>\theta$, $\vec{x}$ 为待分类的样本, $split(\vec{x})$ 为测试函数,在随机森林分类器中,$\theta$通常基于样本的某一特征进行设定,而在极端随机森林分类器中,$\theta$则是随机选择的.

极端随机森林分类器在分类精度和训练时间开销等方面的性能,都要优于随机森林分类器.但极端随机森林分类器只能支持离线训练的方式,不支持增量式学习.为此,本文主要针对解决有限样本的数据流在线学习问题,充分利用极端随机森林分类器的特性,提出了一种支持在线学习的增量式极端随机森林分类器.

## 2 增量式极端随机森林分类器

首先介绍增量式极端随机森林算法的动机和算法描述,再给出算法中重要的分裂阈值参数的相关定理及证明,最后对增量式极端森林分类器的构造开销做出分析.

本文提出的 IERF 算法主要基于以下 3 点考虑:

1) 利用统计量进行增量学习的算法往往不存储样本,而是通过对海量数据进行相关的数据统计分析来获取分类器扩展的必要信息,统计完的样本即被丢弃不再使用,因此在处理小样本数据时,由于不能提供足够的统计信息而使增量学习失败.为此,我们考虑采用存储样本的方式进行增量学习,在后续增量学习过程中仍然能够重复利用历史信息,这样即便对于小样本数据流仍然能够进行稳定的增量学习.

2) 传统的基于单棵决策树的增量学习方法需要保证决策树上所有的决策属性是最优的,而分类器集成(ensemble)的方法,例如极端随机森林[22]算法等,则不需要每棵决策树上所有的决策属性最优,因此利用集成方法可以避免每棵决策树的贪婪重构.同时单棵决策树存在过拟合问题,而集成方法则可以有效地避免此类问题,并能够减少偏差(bias)与方差(variance),同时还能够保持计算的高效性.因此,我们采用分类器集成的方法进行增量学习.

3) 根据样本集合的混乱程度进行节点的分裂,既可以控制树的生长速度,又可以降低对数据噪声的敏感度.

### 2.1 IERF算法

IERF 分类器同 ERF 分类器一样,构造了一个包含多棵决策树的集合,ERF 分类器中每棵决策树称为超树(extra-tree),在 IERF 分类器中,每棵决策树则称为增量超树(incremental extra-tree).每棵增量超树采用 ERF 分类器的思想,基于原始的训练样本集合,采用传统的自上而下的方式进行构造,相对 bootstrap 采样替换策略,这种方法可以减小偏差(bias)[22].构造单棵增量超树的算法见算法 1.

**算法 1**. 构造增量超树($s$,$i$).

输入:一个训练样本 $s$,树索引 $i$.

输出:增量超树 $t_i$.

1. **if** 增量超树 $t_i$ 的根节点不存在
2. 生成一个新的叶节点 $R$,该叶子节点标记值为训练样本 $s$ 的标记值.
3. 返回一个根节点为 $R$ 的决策树 $t_i$
4. **end if**
5. 样本 $s$ 经超树分类后落到叶节点 $L$ 上.
6. 存储样本 $s$ 到叶节点 $L$ 的样本列表 $l$ 中.
7. 更新叶节点 $L$ 上所有样本类别的数量.
8. **if** Gini($L$)>SPLITTHRESHOLD $\Delta$
9. 产生一个新的树节点 $T$,$T$=构造一棵新的子增量超树($l$).
10. **if** $L$ 是根节点,则将 $T$ 作为新的根节点
11. **else** 将 $T$ 替换掉 $L$,作为 $P_L$ 的子节点,其中 $P_L$ 是 $L$ 的父节点.
12. **end if**
13. 删除叶节点 $L$
14. **end if**
15. 返回增量超树 $t_i$.

增量超树中的每个叶节点都维护一个样本列表,用来存储分类到该叶节点上的样本,同时统计当前该叶节点上所有样本标记值的数量分布.当一个新的训练样本到达时,首先经过当前的增量超树分类,到达某一叶节点后被存储到该叶节点上的样本列表中,同时该样本对应的类别标记总数加 1.选择存储样本的原因是在后续过程中可以重复利用这些已获知的样本,用来进行增量超树的构造和扩展,即便是在获得很少样本的情况下仍然可以有效地构造增量超树,同时也避免了计算需要大量样本的 Hoeffding 边界.当叶节点上存储的样本集合到达一定的混乱程度后,即对当前叶节点进行分裂,构造新的决策节点和相应的后继叶节点,从而达到扩展生长整个增量超树的目的.所有增量超树的构造和扩展都是相互独立的.

判断叶节点何时转变成决策节点,我们考虑采用 Gini 系数,它是度量样本集合纯度的一个量(对样本集合纯度度量方法的讨论见第 2.2 节).针对一个样本空间 $D$,其中包含有 $n$ 个样本,共 $k$ 个类别,则

$$Gini(D)=1-\sum_{i=1}^{k}p_i^2,$$

其中,$p_i$ 是类别为 $i$ 的样本在全部样本中所占的比例.当某叶节点上的 Gini 系数超过了设定的阈值Δ时,我们即认为当前叶节点上的样本集合达不到要求的纯度,或被认为样本集合的混乱程度达到给定的上限,这时需要分裂当前叶节点,利用存储的样本构造新的子增量超树.

子增量超树构造过程见算法 2.当有新的叶节点生成时,在当前叶节点上构造一个样本列表,统计并存储样本信息.当一个叶节点分裂时,按照文献[22]的方法来选择一个决策属性和分裂测试,同时构造一个新的决策节点并生成两个相应的后继叶节点.分裂前的叶节点上存储的样本按照相应的决策属性和分裂测试进行划分,得到相应的两个样本集合,分别存储到对应的两个新生后继叶节点上,同时对类别标记的数量重新统计.

随着时间推移,叶节点上存储的样本会逐渐增多,这样既会造成过重的空间负载,又会存在分类器对数据描述不够准确的隐患.为此,我们采用 Least-Recent-Used 替换策略来减少陈旧的样本,着重关注当前的样本.在算法的实现过程中,每个样本被赋予一个时间戳,当样本的时间戳小于给定时刻,即被丢弃.

**算法 2**. 构造子增量超树($S$).

输入:训练样本集合 $S$.

输出:增量超树 $t$

1. **if**:

(1) $|S|<n_{\min}$,$n_{\min}$ 为设定的最小样本数[22]or

(2) $S$ 中所有的候选属性都不变,or

(3) $S$ 中所有的输出变量都一致

2. 生成一个新的叶节点 $t$.
3. 存储 $S$ 中所有样本在节点 $t$ 上.
4. 统计 $S$ 中所有类别标记的数量分布并存储在节点 $t$.
5. 返回叶节点 $t$,$t$ 对应的标记值根据 $S$ 中类别标记数量来确定.
6. **else**:
7. 将 $S$ 划分为两个子集,决策属性和分裂测试 $s^*$根据文献[22]中的方法来选择.
8. 根据子集 $S_l$ 和 $S_r$ 分别构造子树 $t_l$=构造子增量超树($S_l$)和子树 $t_r$=构造子增量超树($S_r$).
9. 根据分裂测试 $s^*$生成决策节点 $t$,将 $t_l$ 和 $t_r$ 分别作为 $t$ 的左子树和右子树.
10. **end if**
11. 返回超树节点 $t$.

### 2.2 样本集合纯度度量方法的讨论

我们定义一个样本集合是纯的,当且仅当该样本集合中大部分元素为同一类样本;定义一个样本集合是不纯的,当且仅当该集合中包含了很多其他类别的样本.如何对样本集合的纯度进行量化度量,常见的方法有3种[24]:Entropy,Gini 系数,以及 Classification error.

$Entropy=\sum_j -p_j\cdot\log_2^{p_j}, Gini=1-\sum_j p_j^2, Classification\ \ error=1-\max\{p_j\}$, 其中这 3 个量都需要测量样本集合中每个类别样本所占的比例 $p_j$(对任意的 $j$ 都存在 $0\leqslant p_j\leqslant 1$).

不论样本集合中样本的种类有多少,Gini 系数均满足 $0\leqslant 1-\sum_j p_j^2\leqslant 1$,但 $Entropy$ 则不存在该性质.例如,假设样本集合中存在 $m$ 个类别的样本($m>2$),且每种类别的样本所占比例均为 $1/m$,即 $p_j=1/m$,则 $Entropy=\sum_m -\frac{1}{m}\log_2^{\frac{1}{m}}=\log_2^m>1$.

相对于 Classification error 而言,Gini 系数考虑了所有类别样本的比例关系,能够反映不同类别样本的比例变化,而 Classification error 只考虑了集合中比例最大的样本,对各类别样本的比例变化有时不能够准确反映.例如,假设样本集合中共含 3 种类别的样本,每种类别样本的比例分别为 0.3,0.4,0.3,当样本集合中各个类别的数量发生了变化,其比例分别变为 0.4,0.2,0.4,则此时 Gini 系数的变化量为 $\Delta Gini=(0.4^2-0.3^2)+(0.2^2-0.4^2)+(0.4^2-0.3^2)=0.02>0$,而 Classification error 的变化量则为 0.

基于以上的比较和讨论,我们选择 Gini 系数作为本文算法中样本集合纯度的度量方法.

### 2.3 分裂阈值参数的选择

本文采用 Gini 系数度量样本集合的混乱程度,当叶节点的 Gini 系数超过了给定阈值时,即对该叶节点进行分裂.如何选择合适的阈值参数,在这里给出详细的讨论.首先引入一个用于度量样本集合中各类别样本比例关系的量——样本混合比 $\alpha$,$\alpha$ 为叶节点上非数量最多的样本相对于数量最多的某一类样本的比例.

**定理 1**. 假设某一叶节点上存储的样本共 $K$ 种,其中数量最多的某类样本标号为 $L_m$,占全部样本数量的比例为 $p_m$,其余样本所占比例为 $p_i(i\neq m, i=1,\ldots,K)$,$L_m$ 样本数量为 $n$,其余类别的样本数目相对 $L_m$ 的比例为 $\alpha_i$,标号非 $L_m$ 的样本相对 $L_m$ 的比例为 $\alpha$, $\alpha=\sum_{i=1,i\neq m}^{K}\alpha_i$,则该叶节点分裂阈值参数 $\Delta=1-\left(\frac{1}{1+\alpha}\right)^2$.

证明:当前叶节点 Gini 系数 $g=1-\sum_{i=1}^{K}p_i^2=1-p_m^2-\sum_{i=1,\ i\neq m}^{K}p_i^2$,

由 $p_m=\frac{n}{n+\sum_{i=1,i\neq m}^{K}\alpha_i\cdot n}=\frac{n}{n+\alpha\cdot n}=\frac{1}{1+\alpha}, p_{i,i\neq m}=\frac{\alpha_i\cdot n}{n+\alpha\cdot n}=\frac{\alpha_i}{1+\alpha}$,得

$$g=1-\left(\frac{1}{1+\alpha}\right)^2-\sum_{i=1,i\neq m}^{K}\left(\frac{\alpha_i}{1+\alpha}\right)^2=1-\left(\frac{1}{1+\alpha}\right)^2-\sum_{i=1,i\neq m}^{K}\left(\frac{1}{1+\alpha}\right)^2\cdot\alpha_i^2=1-\left(\frac{1}{1+\alpha}\right)^2\left(1+\sum_{i=1,i\neq m}^{K}\alpha_i^2\right).$$

又 $\sum_{i=1,i\neq m}^{K}\alpha_i^2\leqslant\left(\sum_{i=1,i\neq m}^{K}\alpha_i\right)^2=\alpha^2$,

$$g=1-\left(\frac{1}{1+\alpha}\right)^2\left(1+\sum_{i=1,i\neq m}^{K}\alpha_i^2\right)\leqslant 1-\left(\frac{1}{1+\alpha}\right)^2(1+\alpha^2)\leqslant 1-\left(\frac{1}{1+\alpha}\right)^2.$$

$\alpha$=0 时,等式成立.故叶节点分裂阈值参数 $\Delta=1-\left(\frac{1}{1+\alpha}\right)^2$. □

对叶节点分裂阈值参数的讨论如下:

(1) 当 $0\leqslant\alpha<1$ 时,$L_m$ 样本的数量一定分别多于其他各个类别样本的数量,否则至少存在一个 $\alpha_l\leqslant 1,l\in[1,K],l\neq m$,这样便有 $\alpha=\sum_{i=1,i\neq m}^{K}\alpha_i\geqslant 1$,与 $0\leqslant\alpha<1$ 矛盾,所以 $L_m$ 样本的数量一定分别多于其他各个类别样本的数量.此时 $L_m$ 在整个叶节点的样本集合中所占比例大于 50%,我们认为此时的叶节点样本集合是纯的.

(2) 当 $\alpha=\sum_{i=1,i\neq m}^{K}\alpha_i\geqslant 1$ 时,不一定存在 $\alpha_l$ 满足 $\alpha_l\geqslant 1,l\in[1,K],\ l\neq m$,即可能任意的 $\alpha_i,\ i\in[1,K],\ i\neq m$ 都小于 1,但当 $\alpha\geqslant 1$ 时,$L_m$ 的样本数量为 $n$,其余非 $L_m$ 样本视为一类,其数目为 $\alpha\cdot n\geqslant n$,$L_m$ 在整个叶节点的样本集合中所占比例 $\frac{n}{n+\alpha\cdot n}\leqslant 50\%$,我们认为此时的叶节点样本集合是不纯的.

综上所述,我们选择 $\alpha$=1 作为叶节点样本集合纯度的临界点,在 $0\leqslant\alpha\leqslant 1$ 中,选择分裂阈值参数. $0\leqslant\alpha\leqslant 1$ 时, $0\leqslant\Delta=1-\left(\frac{1}{1+\alpha}\right)^2\leqslant\frac{3}{4}$, $\Delta$与$\alpha$为单调关系.分裂阈值参数$\Delta$如何选择可以通过$\alpha$值来确定,如何选择适当的$\alpha$值由第 3.1.1 节的实验给出.

当 $\alpha>1$ 时,在仅存两类样本的情况下,另一类样本的数量一定大于 $L_m$ 的数量;而在多于两类样本的情况下,则不一定能够保证存在其他某类样本,其数量大于 $L_m$ 的数量.因此,本文在 $0\leqslant\alpha\leqslant 1$ 中选择分裂阈值参数,相当于适度提早了叶节点的分裂操作,目的在于保证分类器能够更及时反映数据流中样本的特征,增强分类器的判别性能,第 3.1.2 节对此进行了实验证明.

第 3.1.2 节分别采用了两种叶节点分裂判别条件进行对比实验,一种是本文算法采用的在 $0\leqslant\alpha\leqslant 1$ 中选择得到的阈值参数$\Delta$,另一种是采用等待出现非 $L_m$ 的某类样本,其数量大于等于 $L_m$ 样本数量.结果证明,本文的方法能够保证叶节点及时分裂,提高了分类器的判别能力,具有更高的分类正确率和可靠性.

### 2.4 IERF模型构造开销分析

IERF模型构造开销在这里考虑两方面的因素[8]:一是对样本进行遍历统计需要的计算费用,称为实例开销(instance-count additions);另一个是选择决策节点分裂属性时需要的熵信息计算费用(e-score calculations),这种方法需要计算多个属性的信息熵,选择信息熵最大的属性作为分裂属性.

给定$n$个训练样本,其中包含$m$个类别(不妨假设$n>m$),每个样本包含$d$个属性(这里假设所有属性为连续的数值属性).下面对ERF算法和IERF算法的计算开销进行比较(以单棵决策树为研究对象进行讨论).

(1) ERF算法采用离线批量学习的方法,将全部样本一起训练.对$n$个训练样本而言,每棵决策树每层的实例开销为 $n\cdot k$,其中ERF算法默认的参数 $k=\sqrt{d}$;决策树每层熵计算开销为 $\gamma\cdot n\cdot k$,$\gamma$为一正整数常量,取决于熵信息的计算方法.假设得到的决策树高度为$D$,则有:

实例开销 $I=\sum_{i=1}^{D}k\cdot n=D\cdot k\cdot n=D\cdot\sqrt{d}\cdot n$,熵信息计算开销 $E=\sum_{i=1}^{D}k\cdot\gamma\cdot n=D\cdot\sqrt{d}\cdot\gamma\cdot n$,

ERF算法总的计算开销$S_{ERF}$取决于 $I+E=D\cdot\sqrt{d}\cdot(\gamma+1)\cdot n=D\cdot\sqrt{d}\cdot C\cdot n,C=\gamma+1$.

$S_{ERF}$受决策树的高度$D$、样本属性$d$以及熵信息的计算方法三方面因素的影响.

(2) 假设 $n$ 个训练样本按照时间顺序依次到来,IERF 算法每个时刻处理一个样本.新增一个样本时,IERF 算法无须重新学习以往所有样本,而在已有模型基础上进行增量学习即可.新样本首先通过决策树分类(此处时间开销非常小),最终落到某个叶节点 $l$ 上,假设该叶节点 $l$ 已包含 $n_l-1$ 个样本.

由算法 2 知,首先进行叶节点分裂判断,此时无须遍历所有 $n_l$ 个样本,只需根据叶节点上维护的各类别样本的数目进行计算,时间开销非常小.若叶节点分裂条件不满足,则进行下一个样本的处理;若叶节点分裂条件满足,则需要生成新子树,此时的计算开销等同于对 $n_l$ 个样本,利用 ERF 算法批量构造一棵高度为 $D_l$ 树的计算开销.由上述 ERF 算法的计算开销得知,此时 IERF 算法的计算开销为 $D_l\cdot\sqrt{d}\cdot C\cdot n_l, C=\gamma+1$.

由此可知,假定已处理完 $n-1$ 个样本,则处理一个新增样本,IERF 算法开销最大为 $D_l\cdot\sqrt{d}\cdot C\cdot n_l$, $C=\gamma+1$,要远小于 ERF 算法重构全部 $n$ 样本的开销 $D\cdot\sqrt{d}\cdot C\cdot n, C=\gamma+1$.

对于全部 $n$ 个样本,假设 IERF 算法在 $t_1,\ldots,t_p$ 共 $p$ 个时刻进行了叶节点分裂($p<n$),则处理 $n$ 个样本的总计算开销 $S_{IERF}$ 主要取决于 $\sum_{i=t_1}^{t_p}(D_i\cdot n_i\cdot\sqrt{d}\cdot C), C=\gamma+1$,其中,$n_i$ 为第 $i$ 时刻新样本到达的叶节点上样本的个数,$D_i$ 为第 $i$ 时刻该叶节点分裂后生成新子树的高度($D_i<n_i$).$S_{IERF}$ 受到叶节点的样本个数 $n_i$、新生子树高度 $D_i$、样本属性 $d$ 以及熵信息的计算方法等五方面因素的影响.

令 $\phi_{mean}=\dfrac{\sum_{i=t_1}^{t_p}\phi_i}{p}, \phi_i=D_i\cdot n_i$,则有 $S_{IERF}=\sum_{i=t_1}^{t_p}(D_i\cdot n_i\cdot\sqrt{d}\cdot C)=p\cdot\phi_{mean}\cdot\sqrt{d}\cdot C$.

因此,IERF 算法与 ERF 算法相比,处理相同的 $n$ 个样本,若满足 $S_{ERF}-S_{IERF}=n\cdot D\cdot\sqrt{d}\cdot C-p\cdot\phi_{mean}\cdot\sqrt{d}\cdot C>0$,等价于满足 $\dfrac{p}{n}\cdot\dfrac{\phi_{mean}}{D}<1$.第 3.2 节实验结果表明,IERF 算法的时间开销要小于 ERF 算法,针对每个数据集分别给出了 $\varphi_{mean}$,$p$ 及 $D$ 的统计,由结果得知,IERF 分类器叶节点上存储的样本数量较少,因而 $\varphi_{mean}$ 很小,并且叶节点分裂次数 $p$ 有限,上述约束条件容易满足,从而导致 IERF 算法的时间开销通常要小于 ERF 算法.

## 3 机器学习数据集实验

本节基于机器学习数据集进行实验,共分3部分:首先,第3.1节通过实验确定样本混合比$\alpha$与IERF性能的关系,从而选择合适的叶节点分裂阈值参数$\Delta$;其次,第3.2节在分类正确率、模型复杂度以及训练时间开销等方面,对增量学习的IERF算法和离线批量学习的ERF算法进行比较,以验证IERF算法的正确性;最后,第3.3节在适度规模的数据集上,将IERF算法与其他主要的增量学习算法进行对比分析,检验IERF算法的性能.

本节实验采用的机器学习数据集,除了USPS来自LIBSVM数据集以外,其余均来自UCI数据集.表1包括6个数据集,用于第3.1节分裂阈值参数的选择实验;表2包括7个数据集,用于第3.2节和第3.3节的对比实验.表1和表2中的数据集、样本类别包括二值也包括多值,样本属性的范围也较为宽泛,两表中加*的数据集为小训练样本数据集,未加*的为大训练样本数据集,所有的训练样本集规模适度.表1中Waveform2数据带有噪声,其余数据集中的数据不含噪声.为了模拟适度规模的样本数据流,所有UCI数据实验每次均从训练数据集中随机抽取(不放回)一个训练样本供IERF算法学习,直到所有训练样本全部学习完毕.本节实验中,IERF算法不遗忘训练样本.实验在2.3 GHz,1GB PC上进行.

**Table 1** The first dataset list used in the experiment of section 3.1

**表1** 第3.1节实验所用数据集列表

| Dataset | Class | Attribute | Training set size | Test set size |
|---|---|---|---|---|
| Iris* | 3 | 4 | 100 | 50 |
| Wine* | 3 | 13 | 36 | 142 |
| Waveform2* | 3 | 40 | 300 | 4 700 |
| Semeion* | 10 | 256 | 797 | 796 |
| USPS | 10 | 256 | 7 291 | 2 007 |
| Letter | 26 | 16 | 16 000 | 4 000 |

**Table 2** The second dataset list used in the experiments of section 3.2 and 3.3

**表 2** 第 3.2 节和第 3.3 节实验所用数据集列表

| Dataset | Class | Attribute | Training set size | Test set size |
|---|---|---|---|---|
| Spam | 2 | 57 | 3 221 | 1 380 |
| Ionosphere* | 2 | 34 | 300 | 51 |
| Waveform* | 3 | 21 | 300 | 4 700 |
| Vehicle* | 4 | 18 | 761 | 85 |
| Satellite | 6 | 36 | 4 435 | 2 000 |
| Segment | 7 | 19 | 2 079 | 231 |
| Vowel* | 11 | 10 | 891 | 99 |

### 3.1 分裂阈值参数的选择

#### 3.1.1 分裂阈值参数$\Delta$的确定

叶节点分裂阈值参数$\Delta$影响IERF算法的性能,合适的阈值$\Delta$可以保证IERF算法具有高的分类正确率和低的时空开销.由前面所述,分裂阈值参数$\Delta$的选择与样本混合比$\alpha$大小有关,且$\alpha$取值在[0,1]内,为此我们将样本混合比$\alpha$按照0,0.1,…,1.0进行取值,对不同$\alpha$取值下的IERF算法(采用50棵树)分别统计分类错误率、模型复杂度(叶节点数量)和训练时间开销,最后综合考虑各方面因素从而确定合适的阈值参数$\Delta$.

不同$\alpha$取值下的IERF算法在小训练样本数据集上运行50次,在大训练样本数据集上运行10次.选择表1中的6个数据集进行实验.算法每次运行时,每个数据集都被随机划分为一个训练集和一个测试集,各自集合的样本数量划分见表1.

IERF 算法在不同$\alpha$取值下的分类错误率如图 1 所示,当$\alpha\in[0,0.2]$时,所有数据集上的分类错误率低且稳定,当$\alpha>0.2$时,部分数据集的分类错误率开始变大,当$\alpha>0.5$时,所有数据集的分类错误率显著变坏.这是由于$\alpha$过大导致叶节点分裂迟滞,以至于整体分类能力下降,特别在数据集Waveform2,Wine以及Iris上,$\alpha$逼近1时分类器已经完全失效.

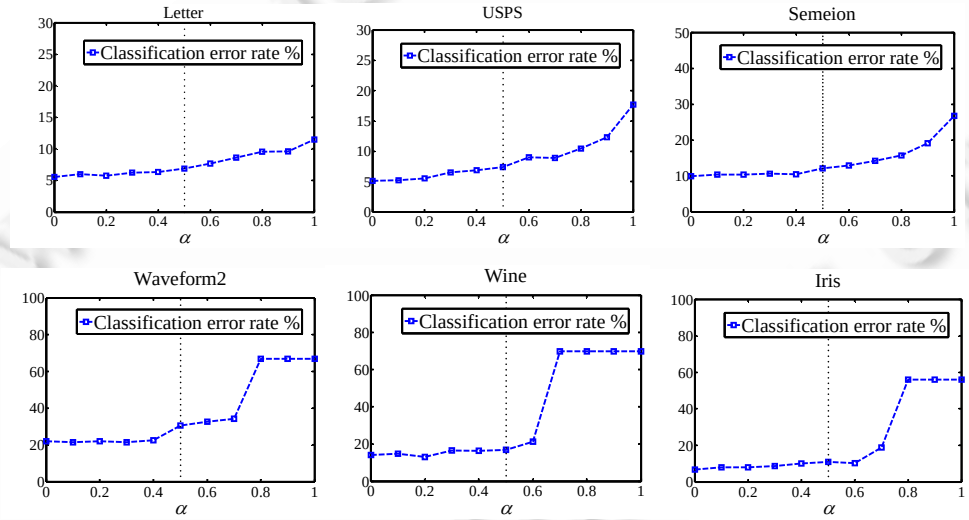



Fig.1 Classification error rates of IERF with different $\alpha$

图 1 不同$\alpha$取值下 IERF 算法的分类错误率

IERF 算法模型复杂度(叶节点的数量)和训练时间开销与样本混合比$\alpha$的关系,分别如图 2 和图 3 所示.由实验结果可以看出,当样本混合比$\alpha$接近 0 时,IERF 算法生成的叶节点数量增多,模型复杂度变大,并且总的训练时间开销增大,这是由于随着$\alpha$减小,每个叶节点的分裂次数会逐渐增多,从而导致 IERF 算法生成的叶节点数量增多,并且总训练时间变大.当$\alpha>0.5$时,由图 2 可以看出,不同数据集下的 IERF 模型严重变坏,特别是图 2 中的 Waveform2,Wine 以及 Iris,这也解释了为何分类器在$\alpha$逼近 1 时完全失效.

综合考虑 IERF 算法的分类错误率、模型复杂度以及训练时间开销等因素,我们选择样本混合比$\alpha$=0.2,介于 0~0.5 之间,使 IERF 算法既具有高的分类正确率,又保持了适当的模型复杂度和低训练时间开销. $\alpha$=0.2 对应

的分裂阈值参数$\Delta$=0.31,我们将此参数作为 IERF 算法使用的默认参数.

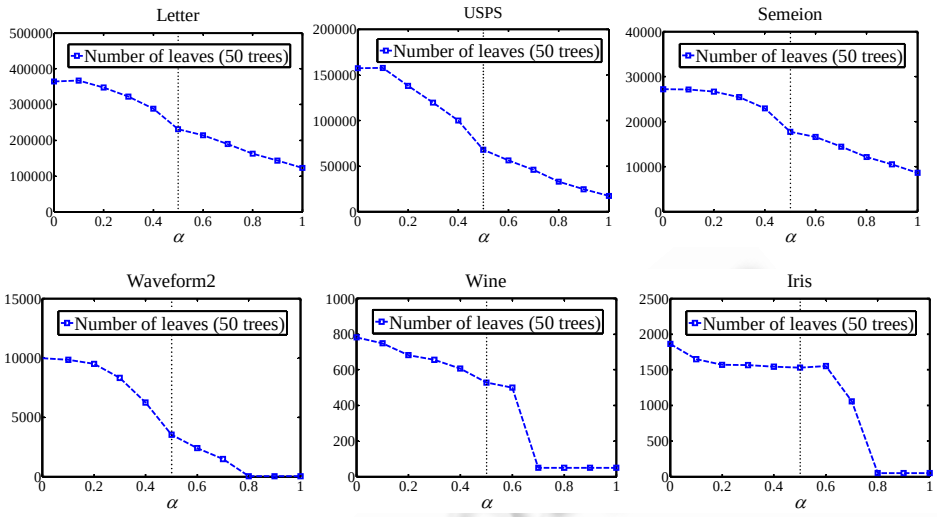



Fig.2 Model complexities of IERF with different $\alpha$

图 2 不同$\alpha$取值下 IERF 算法的模型复杂度

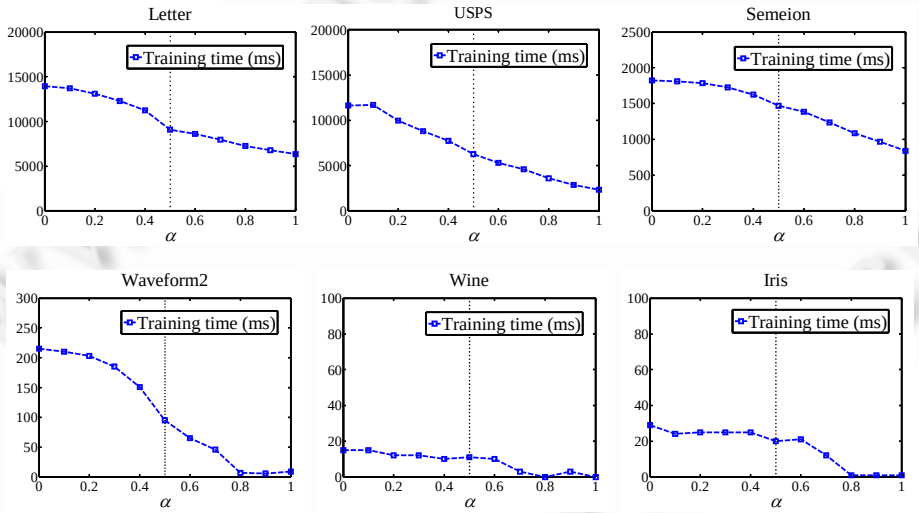



Fig.3 Training time of IERF with different $\alpha$

图 3 不同$\alpha$取值下 IERF 算法的训练时间开销

3.1.2 $\alpha$取值范围合理性的验证

第 2.3 节讨论中指出,在 $0\leqslant\alpha\leqslant1$ 中选择分裂阈值参数,相当于适度提早了叶节点的分裂操作,其目的在于保证分类器能够更及时的反应数据流中样本的特征,增强分类器的判别性能.我们在此对$\alpha$取值范围的合理性进行实验验证.

这里分别采用两种叶节点分裂判别条件进行对比实验,一是采用在 $0\leqslant\alpha\leqslant1$ 中确定的默认参数$\Delta$=0.31 作为叶节点分裂判断依据,称为本文规则(our rule);二是采用等待出现某类非 $L_m$ 的样本($L_m$ 为叶节点分裂前数量最多的某类样本),其数量大于等于 $L_m$ 样本数量,作为对比规则(contrast rule).

按照两种不同的分裂规则,IERF 算法分别在表 1 的小训练样本数据集上运行 50 次,在大训练样本数据集

上运行 10 次,训练集和测试集划分等配置同第 3.1.1 节.我们对两种规则下单棵树平均的叶节点分裂次数、平均的叶节点数量、叶节点上平均存储的样本数量以及整个分类器的分类错误率分别进行了统计,见表 3.

**Table 3** Comparison results between different split rules

**表 3** 叶节点分裂判别条件的比较结果

| Dataset | Average split times per tree | | Average number of leaves per tree | | Average number of stored samples per leaf node | | Classification error rate (%) | |
|---|---|---|---|---|---|---|---|---|
| | Our rule | Contrast rule | Our rule | Contrast rule | Our rule | Contrast rule | Our rule | Contrast rule |
| IRIS | 9 | 3 | 1 561 | 1 047 | 14 | 27 | 9.8 | 22.0 |
| Wine | 6 | **0** | 688 | 50 | 7 | 18 | 14.60 | **69.71** |
| Waveform2 | 136 | 46 | $9.31\times10^3$ | $3.36\times10^3$ | 3 | 20 | 19.11 | 31.13 |
| Semeion | 449 | 197 | $2.66\times10^4$ | $1.38\times10^4$ | 3 | 17 | 10.18 | 19.13 |
| USPS | $2.1\times10^3$ | $0.67\times10^3$ | $1.37\times10^5$ | $0.44\times10^5$ | 35 | 157 | 5.53 | 13.21 |
| Letter | $5.8\times10^3$ | $2.9\times10^3$ | $3.47\times10^5$ | $1.93\times10^5$ | 8 | 29 | 5.75 | 9.58 |

从表 3 可以看出,采用对比规则的 IERF 算法,平均分裂次数远远小于采用本文规则算法.分裂次数的减少,直接导致了每个叶节点上存储的样本数量增多,叶节点的判别力下降;同时,分裂次数的减少也导致了单棵树叶节点数量的减少,使得整个分类器模型的复杂度变小,整体的分类能力降低.表 3 中还可以看出,采用对比规则的算法,分类错误率大大高于采用本文规则的算法,特别是在 Wine 数据集上,对比规则的算法,在根节点上分裂条件始终得不到满足,以至于根节点的分裂操作一再被延迟,最终导致分类器完全失效.

综上所述,本文在 $0\leqslant\alpha\leqslant1$ 中选择分裂阈值参数是正确有效的.

### 3.2 IERF算法与ERF算法的对比实验

为验证IERF算法的正确性,本节将增量学习的IERF算法和批量学习的ERF算法,在表2的Spam,Ionosphere,Waveform,Vehicle,Satellite,Segment和Vowel这7个UCI数据集上进行对比实验,分别比较两者的分类正确率、模型复杂度以及训练时间开销;对第2.4节中给出的IERF算法和ERF算法的部分性能参数进行了统计;最后,在噪声数据的敏感性方面对二者进行了比较.

两种算法在小训练样本数据集上运行50次,大训练样本数据集上运行10次.训练集和测试集的划分见表2.两种算法各使用100棵树.IERF算法分裂阈值参数$\Delta=0.31$,且不遗忘样本,ERF算法使用文献[22]中的默认参数.

#### 3.2.1 非噪声数据实验

从表4可以看出,IERF算法的分类正确率与离线批量学习的ERF算法相当,甚至在有些数据集上优于ERF算法,例如在Ionosphere数据集.由于是增量学习,IERF算法生成的叶节点数量比ERF算法的多,但在相同大小的训练数据集上,IERF算法总的训练时间开销要比ERF算法的小.

**Table 4** Comparison results between IERF and ERF

**表 4** IERF 算法与 ERF 算法的比较结果

| Dataset | Classification accuracy (%) | | Number of leaves | | Training time (s) | |
|---|---|---|---|---|---|---|
| | ERF | IERF | ERF | IERF | ERF | IERF |
| Spam | 95.83 | 95.04 | 51 155 | 59 384 | 16.125 | **3.131** |
| Ionosphere | 92.55 | **94.12** | 7 134 | 8 202 | 0.500 | **0.206** |
| Waveform | 83.39 | 82.42 | 10 782 | 15 919 | 0.583 | **0.312** |
| Vehicle | 74.00 | 71.51 | 29 812 | 42 910 | 1.796 | **0.862** |
| Satellite | 91.57 | 89.29 | 83 659 | 109 954 | 14.619 | **3.125** |
| Segment | 98.23 | 97.72 | 24 051 | 37 493 | 2.901 | **1.081** |
| Vowel | 98.26 | 97.78 | 33 769 | 48 874 | 2.198 | **0.846** |

本文第2.4节对IERF算法和ERF算法的时间开销进行了比较分析,这里给出有关性能参数的数据统计,见表5.表中每组数据都是两个算法运行10次后的平均值,每个参数的含义见第2.4节.

从表5可以看出,$\varphi_{mean}$的值很小,说明IERF分类器叶节点上存储的样本数量较少,由此新生成子树的高度也很小,故二者乘积的均值$\varphi_{mean}$很小.此外,相对于样本总数$n$,IERF算法的叶节点分裂次数$p$有限,因此第2.4节中给

出的约束条件 $\frac{p}{n}\cdot\frac{\phi_{mean}}{D}<1$ 能够得到满足,为此IERF算法的时间开销通常要小于ERF算法.

**Table 5** Analysis on the running time of IERF

**表 5** IERF 算法时间开销的分析

| Dataset | $\varphi_{mean}$ | $p$ | $D$ | $n$ | $\frac{p}{n}\cdot\frac{\phi_{mean}}{D}$ |
|---|---|---|---|---|---|
| Spam | 65 | 296 | 41 | 3 221 | 0.146 |
| Ionosphere | 12 | 54 | 11 | 300 | 0.196 |
| Waveform | 4 | 113 | 12 | 300 | 0.126 |
| Vehicle | 5 | 331 | 14 | 761 | 0.155 |
| Satellite | 18 | 709 | 19 | 4 435 | 0.151 |
| Segment | 12 | 298 | 15 | 2 079 | 0.114 |
| Vowel | 4 | 393 | 16 | 891 | 0.110 |

3.2.2 噪声数据实验

ERF算法对于数据噪声具有良好的稳定性[22],为了测试IERF算法对于噪声的敏感性,这里将IERF算法和ERF算法在加入了噪声的数据集上分别进行了实验.

表2中随机选择3个测试数据集:Vehicle,Waveform和Ionosphere.实验按照噪声级别从0,0.1,…,0.5的顺序,依次给3个数据集的所有样本加入噪声.这里将噪声级别定义为样本属性被噪声干扰的概率[25].噪声按照如下方法添加:计算数据集上相应属性的标准导数$\sigma$,产生满足$N(0,\sigma)$分布的随机数叠加到样本数据对应的属性值上,同时保持样本数据的类别标记值不变.

噪声实验结果如图4所示,从结果可以看出,随着噪声级别的增加,IERF算法的分类错误率保持稳定,在Vehicle和Waveform数据集上分类准确度逼近ERF算法,在Ionosphere数据集上甚至优于ERF算法.实验证明,IERF算法保持了对数据噪声具有良好稳定性的这一性质.

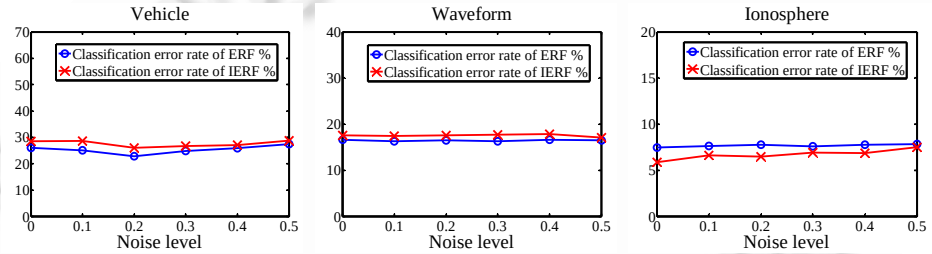



Fig.4 Noise data experiments

图 4 噪声数据实验

### 3.3 IERF算法与其他在线增量学习算法的对比实验

为检验本文提出IERF算法的性能,我们选择了当前几种主要的在线增量学习算法与其比较,包括ITI算法、Online Adaboost 算法和 Online SVM 算法.其中,ITI 算法、以决策树作为弱分类器的 Online Adaboost 算法和 IERF 算法都具有相似的树状结构.4 种在线增量学习算法在表 2 的 7 个数据集上分别进行了实验.

3.3.1 各算法的概述、相应的参数配置与实现情况的说明

ITI 算法是单棵决策树的增量学习算法.每当新的训练样本加入到决策树后,ITI 算法都要检测并修改决策树的决策属性,以确保每个决策节点上决策属性是最优的.修改决策属性的方法在文献[9]中称为属性提升.频繁的属性提升操作对于模型复杂度高的决策树来说将是非常耗时的操作.ITI 算法采用 C 代码实现.

OnlineAdaboost 算法采用广泛使用的 Oza[7]中 Online Adaboost.M1 算法.Oza[7]将数据流按照泊松分布建模,即每个样本数据的到来满足泊松分布.对每次到来的样本数据,每个弱分类器更新 $K$ 次,$K$ 是根据泊松分布 Poisson($\lambda$)产生的随机数.根据新到来的一个样本数据的分类情况来增大或减小泊松分布的参数$\lambda$,进而改变每个弱分类器的更新.为了与 IERF 算法对比,我们选择 100 棵决策树作为弱分类器,决策树更新算法采用 ITI 算法

来实现,整个 OnlineAdaboost.M1 算法采用 C++代码实现.

基于 SVM 的在线增量学习算法有很多,但多数算法只能处理二值分类.ITI 算法、OnlineAdaboost.M1 算法以及本文提出的 IERF 算法都可以处理多类别分类.为了便于比较,我们这里选择文献[4]中给出的可处理多类别分类的 MCSVM 算法.文献[4]中采用迭代的错误边界模型,每次迭代时接收一个新样本,并给出该样本的分类结果,然后根据样本真实标记值更新分类规则.算法使用 C 代码实现,并选择高斯核(sigma=2).

本文提出的 IERF 算法使用 100 棵树,分裂阈值参数$\Delta$=0.31,采用无优化的 C++代码实现.

3.3.2 对比实验

实验使用表 2 中全部 7 个数据集,所有算法在小训练样本数据集上运行 50 次,在大训练样本数据集上运行 10 次.运行时,每个数据集都被随机划分为一个训练集和一个测试集,训练集和测试集的划分见表 2.所有算法都采取如下方法进行训练:每次从训练数据集中随机抽取(不放回)一个训练样本进行学习,直到所有训练样本全部学习完毕.所有算法均不遗忘训练样本.

分类正确率和训练时间开销的结果见表 6 和表 7.从表中可以看出,IERF 算法分类正确率多数情况优于其他算法,并且具有低训练时间开销;以决策树作为弱分类器的 OnlineAdaboost 算法分类性能要优于单棵决策树的 ITI 算法,但训练时间开销非常大,例如表 6 和表 7 中 OlineAdaboost 在大训练样本数据集 Spam 和 Satellite 上运算时间过大,以至于没有给出实验数据,这主要是由于 ITI 算法中单棵决策树的本身运算代价比较大,多棵决策树的更新开销则更显昂贵;Online Multi-SVM 算法可以快速处理多类数据分类的增量学习,时间开销在 4 种算法中最小,但在多数数据集上分类正确率不高,例如 Spam,Ionosphere,Vehicle,Segment 和 Vowel 等数据集上的分类正确率明显低于其他算法.

**Table 6** Classification accuracy of all methods (%)

**表 6** 各方法的分类正确率 (%)

| Dataset | ITI | Online adaboost | MCSVM | IERF |
|---|---|---|---|---|
| Spam | 91.52 | — | 83.41 | **95.04** |
| Ionosphere | 88.23 | 90.98 | 84.31 | **94.12** |
| Waveform | 71.44 | 81.67 | **84.38** | 82.42 |
| Vehicle | 68.23 | **77.88** | 64.71 | 71.51 |
| Satellite | 83.25 | — | 89.25 | **89.29** |
| Segment | **97.83** | 97.66 | 93.91 | 97.72 |
| Vowel | 75.75 | 88.89 | 70.71 | **97.78** |

**Table 7** Training time of all methods (s)

**表 7** 各方法的训练时间开销 (s)

| Dataset | ITI | Online adaboost | MCSVM | IERF |
|---|---|---|---|---|
| Spam | 60.103 | — | 1.391 | 2.931 |
| Ionosphere | 0.629 | 7.818 | 0.016 | 0. 206 |
| Waveform | 2.062 | 9.387 | 0.015 | 0.312 |
| Vehicle | 3.134 | 88.825 | 0.078 | 0.862 |
| Satellite | 235.009 | — | 3.563 | 6.250 |
| Segment | 77.506 | 224.587 | 0.625 | 1.081 |
| Vowel | 40.575 | 679.407 | 0.063 | 0.846 |

## 4 在线视频跟踪

IERF算法可以自然应用于解决在线视频跟踪问题.在视频跟踪实验中,每帧视频图像上用于分类器增量学习的样本数量往往很少,因此,本文提出的IERF算法正好适用于这样的小规模数据流的应用.本节基于真实的视频序列的跟踪实验可以证明,基于IERF算法的在线视频跟踪算法,在复杂环境下能够稳定地跟踪目标物体,并且具备多目标物体跟踪的能力.

首先介绍基于IERF的在线视频跟踪算法,然后给出该算法应用于真实视频序列的实验结果.

### 4.1 基于IERF的在线视频跟算法

我们将IERF算法和协同训练框架[21]结合起来,以实现视频目标的在线跟踪,同时得益于IERF分类器的多类别分类特性,本文的跟踪算法可实现多目标跟踪.具体步骤如下:

(1) 初始化阶段.在第 1 帧视频图像上选择目标区域,提取正负样本训练 IERF 分类器.目标区域的选择可以由用户手动标注完成,也可以通过物体识别算法自动识别完成.包含目标物体的矩形区域作为正样本区域,之外的部分被视为负样本区域.大小规则的图像块作为样本进行处理,相对于单个像素点而言,图像块包含更丰富的信息,并且能够使 IERF 分类器具有快速的学习和识别速度.在初始化阶段,为了给 IERF 分类器提供足够的训练样本,我们在正样本区域和负样本区域内随机产生大量子窗口,这些子窗口可以相互重叠,且位置随机,每个子窗口从 9×9 像素大小到整幅图像大小中随机产生.子窗口对应的图像块即可作为样本进行处理,第 1 帧上 100 个左右的随机子窗口采样得到的图像块即可有效完成分类器的初始化.

(2) 在线增量学习阶段.采用半指导学习理论中的协同训练框架,为 IERF 分类器提供大量无标记样本以完成增量学习.本文采用颜色直方图和梯度方向直方图(HoG)[26]分别作为两个特征空间(可替换为其他类型的特征),在两个特征空间下分别训练两个 IERF 分类器.利用滑动窗口方法,这两个分类器分别对同一帧视频图像进行分类识别.颜色空间的分类器提交一部分自身置信度高的样本交给 HoG 空间的分类器进行增量学习,同时 HoG 空间的分类器也提交一部分自身置信度高的样本给颜色空间分类器进行增量学习.

(3) 目标跟踪阶段.两个特征空间的 IERF 分类器各自产生一幅置信图,两幅置信图基于两个分类器的权值进行叠加整合,最终得到一帧置信图.通过整合的置信图,利用 camshift 算法最终可确定目标物体的位置.

### 4.2 真实视频实验结果

本节给出不同视频序列下的跟踪结果.IERF分类器处理的样本采用 9×9 像素大小的图像块.每个图像块上提取8个方向的HSL颜色直方图作为颜色特征,同时提取包含9个方向bins的HoG特征.每个特征空间下的IERF分类器各自包含30棵树.为了释放不断增长的存储空间,IERF算法选择样本丢弃,这里采用最近使用策略,$T$时刻之前的帧数据被丢弃,根据目标物体和环境的变化情况,$T$选择10~50.

图5的视频序列来自PETS2001,结果证明了本文方法具有多目标跟踪的能力.图6、图7为fragrtrack方法[27]和本文方法的对比结果.图6中的视频序列Rotating Girl来自Stan Birchfield的头部视频数据集,视频中人物头部快速频繁地转动,导致目标外观快速频繁的变化;图7中的视频序列Sylvester来自David Ross的跟踪视频数据集,由于受到环境光照强烈的影响,以及目标物体大尺度的运动,目标外观发生了显著变化.由图6、图7可见,在上述复杂情况下,fragrtrack方法对跟踪目标的更新不够准确,从而导致跟踪失败;而本文方法能够及时学习并更新对目标物体的描述,因此能够准确跟踪.

(a) Initial frame (b) Frame-25 (c) Frame-40 (d) Frame-64

(a) 初始帧 (b) 第 25 帧 (c) 第 40 帧 (d) 第 64 帧

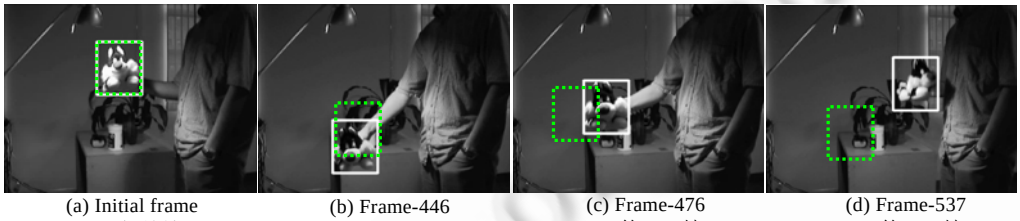
Fig.5 Tracking result pedestrian and car sequences from PETS2001, both the pedestrian and the car were tracked by our approach

图 5 PETS2001 行人与汽车视频序列的跟踪结果.本文算法同时跟踪了行人和车辆

(a) Initial frame (b) Frame-117 (c) Frame-137 (d) Frame-200
(a) 初始帧 (b) 第 117 帧 (c) 第 137 帧 (d) 第 200 帧

Fig.6 Tracking result of Ref.[27] (dot) and our method (line) on head tracking sequence

图 6 文献[27](点状矩形区域)和本文算法(线状矩形区域)的头部视频序列跟踪结果



(a) Initial frame (b) Frame-446 (c) Frame-476 (d) Frame-537
(a) 初始帧 (b) 第 446 帧 (c) 第 476 帧 (d) 第 537 帧

Fig.7 Tracking result of Ref.[27] (dot) and our method (line) on Sylvester sequence

图 7 文献[27](点状矩形区域)和本文算法(线状矩形区域)的 Sylvester 视频序列跟踪结果

表8中给出了上述3个视频测试序列下方法[27]和本文方法的统计结果,正确跟踪的帧数由手工统计完成.如果被目标物体有25%以上没能被跟踪,那么这帧数据将被视为跟踪错误.针对以上全部实验,本文算法通过各视频序列的第1帧数据进行初始化,后续跟踪全部依赖IERF分类器的在线增量学习.实验结果证明了本文提出的IERF算法的可靠性和稳定性.本文的跟踪算法采用无优化C++代码实现,所有实验均在2.3 GHz,1GB PC上完成.针对 320×240 大小的彩色视频图像,本文算法处理速度为5fps.

**Table 8** Tracking results of all methods

**表 8** 各方法目标跟踪实验结果

| Video sequence | Object | Ref.[27] | Our method |
|---|---|---|---|
| Car and girl | Multiple object | — | 67/67 |
| Head sequence | Single object | 403/500 | 500/500 |
| Sylvester sequence | Single object | 467/600 | 600/600 |

## 5 结论与展望

本文提出了一种支持在线学习的增量式极端随机森林分类器IERF.该算法在叶节点上存储样本,并通过计算叶节点上所有样本的Gini系数来判断叶节点是否需要分裂.针对有限数量的样本,特别是小样本数据流的情况,本文算法能够对分类器进行快速有效的扩展.实验首先验证了IERF算法模型复杂度适当,能达到和离线批量学习的ERF算法相当的分类正确率,并且训练时间开销小于ERF算法,同时对噪声数据具有良好的稳定性.其次,验证了在适度规模的数据集合上,与当前主要的增量学习算法相比,本文算法在分类正确率和训练时间开销等方面具有明显优势.最后将IERF算法应用于小样本数据流的一个应用——视频在线跟踪问题,多个真实视频实验验证了基于IERF的在线视频跟踪算法具有良好的稳定性,并且具备多目标跟踪的能力.

本文提出的 IERF 分类器中多棵树的学习和分类完全是独立的,具备完全并行化处理的能力.目前本文算法仍按照串行处理来实现,因此算法的时间开销还具有进一步降低的空间.

**References**:


[1] Schlimmer JC, Fisher DH. A case study of incremental concept induction. In: Proc. of the 5th National Conf. on Artificial Intelligence Philadelphia. Morgan Kaufmann Publishers, 1986. 496−501. https://www.aaai.org/Papers/AAAI/1986/AAAI86-083.pdf

[2] Quinlan JR. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.

[3] Cauwenberghs G, Poggio T. Incremental and decremental support vector machine learning. Neural Information Processing Systems, 2000. 409−415. http://www.cs.cmu.edu/Web/Groups/NIPS/00papers-pub-on-web/CauwenberghsPoggio.ps.gz

[4] Crammer K, Singer Y. Ultraconservative online algorithms for multiclass problems. Journal of Machine Learning Research, 2003,3: 951−991.

[5] Xiao R, Wang JC, Sun ZX, Zhang FY. An incremental SVM learning algorithm $\alpha$-ISVM. Journal of Software, 2001,12(12): 1818−1824 (in Chinese with English abstract). http://www.jos.org.cn/ch/reader/view_abstract.aspx?flag=1&file_no=20011211&journal_id=jos

[6] Ye N, Li XY. A machine learning algorithm based on supervised clustering and classification. In: Proc. of the 6th Int'l Computer Science Conf. on AMT 2001. LNCS 2252, 2001. 327−334. http://www.springerlink.com/content/vlt36tngcauuaruq/

[7] Oza NC, Russell S. Online bagging and boosting. In: Proc. of the 8th Int'l Workshop on Artificial Intelligence and Statistics. 2001. 105−112. http://www.cs.berkeley.edu/~oza/papers/aistats01.ps

[8] Utgoff PE. Incremental induction of decision trees. Machine Learning, 1989,4:161−186. [doi: 10.1023/A:1022699900025]

[9] Utgoff PE, Berkman NC, Clouse JA. Decision tree induction based on efficient tree restructuring. Machine Learning, 1997,29:5−44.

[10] Luo B, Zhou ZH, Chen ZQ, Chen SF. Induce: An incremental decision tree algorithm. Journal of Computer Research and Development, 1999,36(5):518−522 (in Chinese with English abstract).

[11] Domingos P, Hulten G. Mining high-speed data streams. In: Proc. of the Int'l Conf. on Knowledge Discovery and Data Mining. 2000. 71−80. http://www.cs.washington.edu/homes/pedrod/papers/kdd00.pdf

[12] Gama J, Rocha R, Medas P. Accurate decision trees for mining high speed data streams. In: Proc. of the ACM SIGKDD 2003. 2003. 523−528. http://magna.cs.ucla.edu/~hxwang/stream/gama-kdd03.pdf

[13] Kirkby R. Improving hoeffding trees [Ph.D. Thesis]. Department of Computer Science, University of Waikato, 2007.

[14] Wang T, Li ZJ, Hu XH, Yan YJ, Chen HW. An incremental fuzzy decision tree classification method for data streams mining based on threaded binary search trees. Chinese Journal of Computers, 2007,30(8):1244−1250 (in Chinese with English abstract).

[15] Abdulsalam H. Streaming random forests [Ph.D. Thesis]. Kingston: Queen's University, 2008.

[16] del Campo-Ávila J, Ramos-Jiménez G, Gama J, Morales-Bueno R. Improving prediction accuracy of an incremental algorithm driven by error margins. Intelligent Data Analysis, 2008,12(3):305−318.

[17] Friedman N, Goldszmidt M. Sequential update of Bayesian network structure. In: Proc. of the 13th Conf. on Uncertainty in Artificial Intelligence. 1997. 165−174. http://www.cs.huji.ac.il/~nir/Papers/FrG4.pdf

[18] Okamato K, Ozawa S, Abe S. A fast incremental learning algorithm of RBF networks with long-term memory. In: Proc. of the Int'l Joint Conf. on Neural Networks. 2003. 102−107. http://www2.kobe-u.ac.jp/~ozawasei/pub/ijcnn03a.pdf

[19] Witten IH, Frank E. Data mining: Practical machine learning tools and techniques. Morgan Kaufmann Publishers, 2005.

[20] Tang F, Brennan S, Zhao Q, Tao H. Co-Tracking using semi-supervised support vector machines. IEEE Int'l Conf. on Computer Vision. 2007. 1−8. [doi: 10.1109/ICCV.2007.4408954]

[21] Zhu XJ. Semi-Supervised learning literature survey. Madison: Department of Computer Sciences, University of Wisconsin, 2007.

[22] Geurts P, Ernst D, Wehenkel L. Extremely randomized trees. Machine Learning, 2006,63:3−42. [doi:10.1007/S10994-006-6226-1]

[23] Breiman L. Random forests, Machine Learning, 2001,45:5−32.

[24] Breiman L, Friedman JH, Olshen RA, Stone CJ. Classification and Regression Trees. Monterey: Wadsworth and Brooks, 1984.

[25] Hong Y, Kwong S, Chang YC, Ren QS. Unsupervised data pruning for clustering of noisy data. Knowledge-Based Systems, 2008,21:612−616. [doi: 10.1016/j.knosys.2008.03.052]

[26] Dalal N, Triggs B. Histograms of oriented gradients for human detection. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition. 2005. 886−893. http://lear.inrialpes.fr/pubs/2005/DT05/cvpr2005_talk.pdf

[27] Adam A, Rivlin E, Shimshoni I. Robust fragments-based tracking using the integral histogram. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition. 2006. 798–805.

## 附中文参考文献

[5] 萧嵘,王继成,孙正兴,张福炎.一种 SVM 增量学习算法$\alpha$–ISVM.软件学报,2001,12(12):1818–1824. http://www.jos.org.cn/ch/reader/view_abstract.aspx?flag=1&file_no=20011211&journal_id=jos

[10] 骆斌,周志华,陈兆乾,陈世福.一个增量式判定树学习算法 INDUCE.计算机研究与发展,1999,36(5):518–222.

[14] 王涛,李舟军,胡小华,颜跃进,陈火旺.一种高效的数据流挖掘增量模糊决策树分类算法.计算机学报,2007,30(8):1244–1250.

**王爱平**(1981－),男,内蒙古包头人,博士生,主要研究领域为计算机视觉,机器学习.

**程志全**(1977－),男,博士,主要研究领域为计算机图形学.

**万国伟**(1982－),男,博士生,主要研究领域为视频处理,三维重建.

**李思昆**(1941－),男,教授,博士生导师,CCF 高级会员,主要研究领域为虚拟现实与可视化.