ISSN 1004-9037,CODEN SCYCE4
Journal of Data Acquisition and Processing Vol. 33,No. 6,Nov. 2018,pp. 1021－1033
DOI:10.16337/j.1004-9037.2018.06.010

http://sjcj.nuaa.edu.cn
E-mail:sjcj@nuaa.edu.cn
Tel/Fax:＋86-025-84892742

基于集成分类器的数据流分类算法

韩东红　马宪哲　李莉莉　王国仁

(东北大学计算机科学与工程学院，沈阳，110819)

摘　要：作为一种典型的大数据，数据流具有连续、无限、概念漂移和快速到达等特点，因此传统的分类技术无法直接有效地应用于数据流挖掘。本文在经典的精度加权集成(Accuracy weighted ensemble，AWE)算法的基础上提出概念自适应快速决策树更新集成(Concept very fast decision tree update ensemble，CUE)算法。该算法不仅在基分类器的权重分配方面进行了改进，而且在解决数据块大小的敏感性问题以及增加基分类器之间的相异性方面，有明显的改善。实验表明在分类准确率上，CUE 算法高于 AWE 算法。最后，提出聚类动态分类器选择(Dynamic classifier selection with clustering，DCSC)算法。该算法基于分类器动态选择的思想，没有繁琐的赋权值机制，所以时间效率较高。实验结果验证了 DCSC 算法的有效和高效性，并能有效地处理概念漂移。

关键词：数据流；基分类器；集成分类器；决策树；概念漂移；聚类

中图分类号：TP181　　**文献标志码**：A

Data Flow Classification Algorithm Based on Integrated Classifier

Han Donghong，Ma Xianzhe，Li Lili，Wang Guoren

(School of Computer Science and Engineering，Northeastern University，Shenyang，110819，China)

Abstract：As a typical big data，data stream has the features of continuous，infinite，concept drift and fast arrived. The features make it impossible to apply traditional classification techniques to classify data streams. The paper proposes the concept very fast decision tree(CVFDT) update ensemble(CUE) algorithm based on the classic accuracy weighted ensemble (AWE) algorithm. This algorithm not only improves the weight distribution of the base classifier，but also improves the sensitivity of the block size and the increase of the dissimilarity between base classifiers. Experiments show that，in the classification accuracy，CUE algorithm is higher than the AWE algorithm. Finally，the dynamic classifier selection with clustering (DCSC) algorithm is proposed，which is based on the idea of classifier dynamic selection. The time efficiency is relatively high because there is no tedious weight value mechanism. Experimental results show that the DCSC algorithm can effectively handle the concept of drift and its efficiency is relatively high.

Key words：data streams；base classifier；ensemble classifier；decision tree；concept drift；clustering

---

基金项目：国家自然科学基金(61173029，61272182，61672144，61332006)资助项目。
收稿日期：2018-05-25；**修订日期**：2018-07-03

引 言

近年来,数据流作为一种典型的大数据类型得到广泛的应用,例如应用于传感器网络[1]、监控系统[2]以及网络入侵检测[3]等领域。和传统的静态数据不同,数据流的连续、无限、概念漂移和快速等特点使得使用传统的分类技术进行数据流挖掘效果不佳。因此,面向数据流的实时动态更新的分类算法研究成为相关领域的研究热点。

文献[4]使用描述项的值来描述概念,通过向前滑动窗口来删除或者增添描述项。由于该系统存储所有值对,因此该系统具有较高的更新成本,不能实时处理数据流。Wang 等[5]提出了一个精度加权集成(Accuracy weighted ensembles, AWE)模型,用来挖掘概念漂移数据流,用组合的加权分类器来进行数据流挖掘,该模型能够很好地处理概念漂移,并且其分类准确率较高。Domingos 等[6]提出了一种基于 Hoeffding 的快速决策树(Very fast decision tree, VFDT)算法,通过考察将当前的最佳属性作为中间节点,将所使用的测试数据项的数量在 Hoeffding 的边界统计范围之内作为考察依据,但该算法不能解决概念漂移问题。文献[7]在 VFDT 的基础上提出概念自适应快速决策树算法(Concept very fast decision tree, CVFDT),解决了数据流概念漂移问题。文献[8]提出任意窗口流建模算法用于在传感器网络中发现感兴趣的模式。文献[9]在按需分类中采用了 CluStream 的微集群思想,获得了较高的分类准确率。文献[10]提出了一种新分类方法,研究怎样使用历史数据来进行数据流分类。当输入一些新的数据,通过比较以下 4 个分类器的准确性,从中选择准确率较高的分类器:(1) 一个旧的分类器;(2)从新的数据中学习的分类器;(3)从新数据中选择和旧数据相似的数据学习到的分类器;(4)从旧数据中选择和新数据相似的数据学习到的分类器。对这 4 个分类器进行精度检验,选出精度最高的分类器,并将其作为新的分类器。Zhang 等[11]提出了一种双集成模型对偏斜数据流进行分类。文献[12]提出了一种有效的半监督框架,利用检测分类器置信度来观察概念漂移。Osojnik 等[13]利用多目标回归提出了一种新的面向数据流的多标签分类方法。文献[14]针对重现概念漂移检测中的概念表征和分类器选择问题,提出基于主要特征抽取的概念聚类和预测算法。

本文从分析数据流的特点入手,针对具有概念漂移的数据流分类问题,提出一种新颖的基于集成分类模型的数据流分类算法,目的是提高分类精度和时空效率。本文的贡献点如下:

(1) 提出概念自适应快速决策树更新集成(CVFDT update ensemble, CUE),旨在增加集成模型中基分类器的不相似度。首先利用概念自适应快速决策树 CVFDT 在每个数据块上训练决策树,并将训练得到的决策树作为基分类器,循环这个过程,当基分类器的数量达到 L 时停止循环。其次整合每个基分类器通过加权求平均的方法得出最终结果,并将其作为待分类实例的最终分类标签。然后当发生漂移时,导致存在一些不适合当前概念的基分类器,这时将使用最新数据块对其更新。数据块在更新之前被装袋,从而得到各不相同的副本,这使得基分类器之间的不相似性增加。当完成对基分类器进行更新时,要进行对基分类器调整权重的操作。实验表明,该算法的准确率较高。

(2) 提出基于聚类的集成分类算法(Dynamic classifier selection with clustering, DCSC),该算法源于集成分类器的动态选择思想。首先对数据块进行聚类并保存聚类信息。其次训练决策树并将其作为基分类器。并且集成模型保证有 L 个簇信息和基础分类器。然后在对实例进行分类时,分类依据是选择与该实例具有最高相似度的基分类器,该实例的最终类标签就是其分类结果。在集成模型中修剪选择较少或精度低于设定阈值的基分类器,借助新到来的数据训练新的基分类器,并将其加入到集成模型中。最后的实验结果表明,DCSC 处理数据流时具有较高的效率,对突发漂移具有较快的适应性。

1 带有基分类器更新的集成分类算法

1.1 算法描述

CUE 算法是本文提出的一种带有基分类器更新的数据流分类算法,该算法包含集成分类模型的构

造和集成分类模型的更新两部分。

1.1.1 集成分类模型的构造

CUE 算法利用 CVFDT 算法训练基分类器,通过使用后续数据块中的数据对它进行更新。

假设用有序数据块 $S_1, S_2, \cdots, S_n$ 来表示到来的数据流。集成模型在初始时是空的,伴随着数据块 S_1 流入,在 S_1 上使用 CVFDT 训练基础分类器 C_1,并且把训练所得到的基分类器加入到集成模型中。接着数据块 S_2 到来,相应的基础分类器 C_2 被训练并加入到集成模型中。当基分类器的数量满足小于 L 的条件时,给分类器设置权重 1。循环上面的过程,基分类器的数量达到 L 时终止循环。本文提出的集成分类模型在构造阶段不对数据流进行分类,基分类器构造描述如算法 1 所示。

算法 1 基分类器构造算法

输入:S(一个数据流实例)

d (数据块 S_i 的大小)

L(基分类器的数量)

输出:E(L 个在线分类器的集合)

算法描述:

$E \leftarrow \varnothing$

repeat

train classifier C_i on S_i;

add C_i to E and set weight$_i$ = 1;

until L classifiers in E

return E

1.1.2 集成分类模型的更新

CUE 在分类过程中引入了分类器更新机制,该更新机制使得数据块大小对算法性能的影响不再明显,而且通过使用不同的数据更新基分类器,增加了基分类器的不相似程度,使得集成模型在分类性能评估上得到大大的提升。

通过 1.1.1 节对基分类器的训练,集成模型中已经存在训练好的基分类器 L 个。可以认为最新的数据块 S_n 最接近当前数据流中类分布,因此把最新的数据块 S_n 当做测试集,使用式(1)计算所有的基分类器的均方误差 MSE 为

$$\mathrm{MSE}_i = \frac{1}{|S_n|} \sum_{(x,c) \in S_n} (1 - f_c^i(x))^2 \tag{1}$$

式中:数据块 S_n 中的实例形式是(x, c),x 为实例的属性,c 为实例的实际类标签。$f_c{}^i(x)$代表基本分类器 C_i 将实例的类标签贴为 c 的概率,计算数据块 S_n 中的所有实例的 $(1 - f_c^i(x))^2$ 值,并且求和取平均以获得基本分类器 C_i 的均方误差 MSE_i,并且根据 MSE_i 给 C_i 分配权重,则

$$w_i = \frac{1}{\mathrm{MSE}_i + \varepsilon} \tag{2}$$

式中:ε 用来避免 MSE_i 为 0 的情况,将其设置为无穷小正常数。

计算任何一个基分类器对数据块 S_n 中实例的类别进行随意猜测的均方误差 MSE_r 为

$$\mathrm{MSE}_r = \sum_c p(c)(1 - p(c))^2 \tag{3}$$

其中 $p(c)$ 代表在 S_n 中各个类所占比例,其取值在 0 到 1 之间。

CUE 选择 $\mathrm{MSE}_i > \mathrm{MSE}_r$ 的基分类器进行更新,其目的是能够充分利用数据并增加基分类器之间的不相似度。可以理解为:MSE_r 表示任意分类器随机猜测 S_n 中的实例的随机错误概率。当 C_i 的均方误差 MSE_i 大于 MSE_r 时,基分类器对集成模型不起作用,需要进行更新。然而,直接使用 S_n 作为训练数

据更新基分类器将减少基分类器之间的差异性。因此需要先对 S_n 进行装袋操作，得到原始数据块的不同副本，从而增加基分类器彼此间的不相似度和独立性，该集成模型的分类能力也得到提高。算法描述如算法 2 所示。

算法 2 CUE 更新算法

输入：S（一个数据流实例）

d（数据块 S_i 的大小）

k（基分类器的数量）

输出：E（由 k 个可更新权重的在线分类器构成的集成分类器）

算法描述：

for all classifiers $C_i \in E$ do

apply C_i on S_n to derive MSE_i

derive weight w' for C_i using equation (2)

end for

for all classifiers $C_i \in E$ do

if $\mathrm{MSE}_i > \mathrm{MSE}_r$ then

update classifier C_i with S_n using bagging

end if

end for

$E \leftarrow$ updated weighted classifiers $\in E$

classify a new instance with E using weighted average

1.2 实验结果分析

集成分类算法 AWE[5] 具有较好的分类准确率和概念漂移处理能力，是有代表性的数据流分类算法。在本文的实验中，分别在时间效率、内存使用情况和窗口大小对算法的影响及准确率等方面对比本文提出的 CUE 算法和 AWE 算法的差异。在本实验中，AWE 的基分类器通过训练 VFDT 得到。

1.2.1 数据集

采用两个数据集模拟数据流环境来评估 CUE 算法的性能，即美国联邦森林覆盖类型数据集和波形数据集。

(1) 美国联邦森林覆盖类型数据集：它是真实的数据集，该数据源于美国林业局的区域资源信息系统，该数据集共包含数据记录 581 012 条。选择其中的 50 000 条用于实验，每条记录由 54 个属性组成。

(2) 波形数据集：该数据集是一个由 3 个类别标签组成的人工合成数据集，其中每个实例有属性值是实数的属性 40 个，后 19 个有噪声，带有缓慢的概念漂移。

1.2.2 实验分析

两种算法中参数设置如表 1 所示。随着决策树的增长，会产生更多的中间节点，每个中间节点都有一棵替代的子树。存储和处理这些子树也需要很多时间，因此限制树的深度有助于提高算法的效率。这个实验中，限制决策树的最大深度为 8。

表 1 默认实验参数设置

Tab. 1 Default parameter setting

参数	含义	设定值
Base_num	基分类器个数	15
Chunk_size	数据块大小	1 000
δ	分裂置信度	0.01
Recordpoint	观测点	5 000
Tree_height	决策树的深度	8
Grace_period	分裂一个节点最少的数据量	200
τ	打破最佳属性选择阈值	0.05
Window_size	CVFDT 中的滑动窗口	1 500

(1) 时间效率

算法所使用的时间分为训练时间和测试时间。图 1 和图 2 分别显示了 AWE 算法和 CUE 算法分别在两个数据集上训练的平均训练时间。CUE 算法比 AWE 算法的训练时间长。经过分析原因有两个:一方面 CUE 中学习基分类器算法采用 CVFDT;另一方面当 CUE 算法发现集成模型中存在不适应当前概念的基分类器时,使用 Bagging 技术对多个副本进行采样再更新基本分类器。但 AWE 仅使用 VFDT 学习基分类器,和 CUE 算法相比缺少更新分类器和抽取样本两个过程。

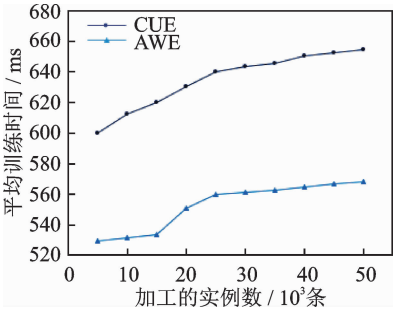

图 1 在波形数据集上一个数据块的平均训练时间对比

Fig. 1 Comparison of average chunk training time on Waveform data set

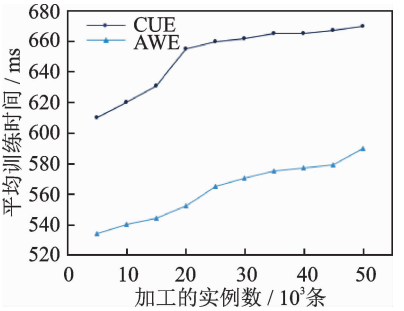

图 2 在森林覆盖数据集上一个数据块的平均训练时间对比

Fig. 2 Comparison of average chunk training time on Forest CoverType data set

图 3 和图 4 显示了分别在波形数据集和森林覆盖数据集上 CUE 和 AWE 算法的测试时间。从图 3,4 中可以看出,CUE 的平均测试略高于 AWE。这是因为 CVFDT 算法的中间节点会随新数据流入而改变,增加了时间成本。另外,两种算法的测试时间不随数据量的增加而增加,维持在一个恒定范围内,这满足数据挖掘对时间的约束。

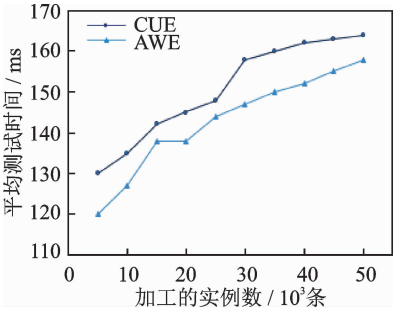

图 3 在波形数据集上一个数据块的平均测试时间对比

Fig. 3 Comparison of average chunk test time on Waveform data set

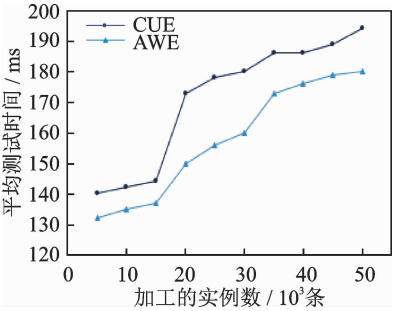

图 4 在森林覆盖数据集上一个数据块的平均测试时间对比

Fig. 4 Comparison of average chunk test time on Forest CoverType data set

(2) 内存占用情况

图 5 和图 6 分别显示了在波形数据集和森林覆盖数据集上 CUE 和 AWE 算法的占用内存情况。可以看出,CUE 消耗的内存比 AWE 多。CVFDT 随着数据增多逐渐增长,特别是当数据集的属性越多,可以替代的子树和中间节点就越多,则需要更多的内存来存储这些子树。另外,CUE 在基分类器的更新过程中提供装袋抽样,需要额外内存存储副本。随着数据量的增加,CUE 能够符合数据流挖掘的内存约束,因为 CUE 算法在运行过程中占用的内存接近于一个常数。

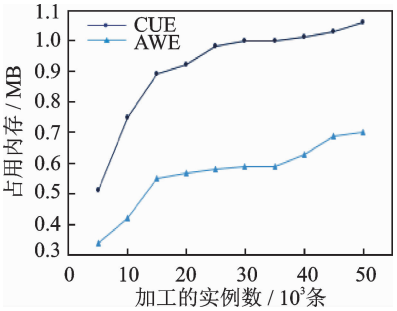

图 5 在波形数据集上内存使用情况对比

Fig. 5 Comparison of memory usage on Waveform data set

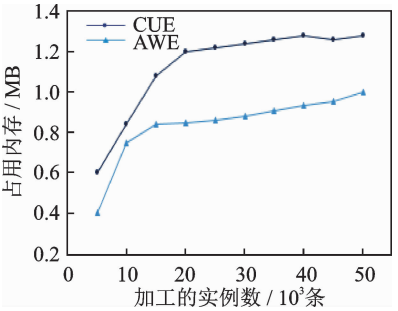

图 6 在森林覆盖数据集上内存使用情况

Fig. 6 Comparison of memory usage on Forest CoverType data set

(3) 准确率

图 7 显示了在波形数据集上两种算法的准确率情况。从图 7 中可以看出 CUE 算法的平均准确率高于 AWE 算法。在 20 000 数据点处,波形数据集产生突变概念漂移,此时 CUE 算法和 AWE 算法的准确率都突然下降。接下来准确率再次提升,且 CUE 有较快的提升速度。

为了形成突变漂移概念,从森林覆盖数据集的开头和中间提取连续的 15 000 条数据,尾部提取 20 000条数据进行实验,结果如图 8 所示。从图 8 中可以看出,两种算法的准确度在加工数据在 15 000 条时开始迅速下降,然后 CUE 准确率逐渐升高,30 000 条数据时再次急剧下降,然后再升高。由于 CUE 适应当前新概念,其准确率总是高于 AWE,并可以快速适应突变概念漂移。

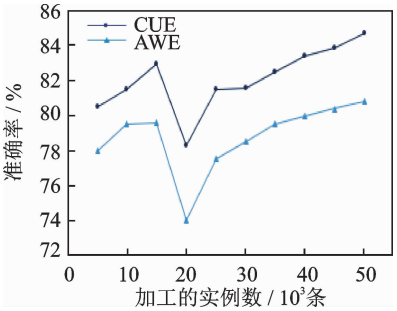

图 7 在波形数据集上的准确率对比

Fig. 7 Comparison of accuracy on Waveform data set

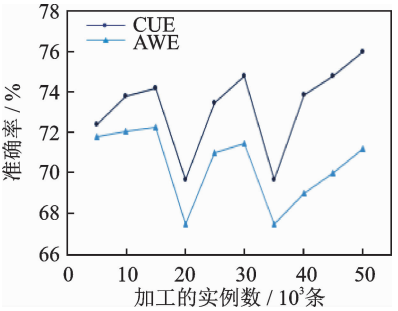

图 8 在森林覆盖数据集上的准确率对比

Fig. 8 Comparison of accuracy on Forest CoverType data set

(4) 数据块大小对算法性能的影响

图 9 和图 10 分别显示了在波形数据集和森林覆盖数据集上数据块的大小对准确率的影响。如图 9 所示,随着数据块增大,在波形数据集上 CUE 算法和 AWE 算法的准确率呈下降的趋势,在1 000处性能最好。主要是因为两种算法在数据块为 500 时都能很快适应流中变化,但由于 AWE 算法的训练数据不足,基分类器的分类能力不强,因此整体分类准确率不高。CUE 算法通过不断使用新数据更新基分类器,几乎不受数据块大小影响,因此整体分类准确率高于 AWE 算法。当数据块大小为1 000时,AWE 算法由于训练数据充分,其准确率大大提升。可见这两种算法在数据块大小为 1 000 时获得最佳的分类准确率。此后随着数据块大小的增加,准确率缓慢下降。这是由于:数据块过大会影响算法对流中变化的反应速率并且使得同一数据块中包含数据存在多种概念,这将导致集成分类器的分类能力不强,学习到的基分类器的分类规则可能模糊等问题。同时从图 9 可以看出,当数据块的大小为

2 500 时,CUE 算法比 AWE 算法的准确率低。数据块较大的情况掩盖了 CUE 算法基分类器可更新的优点。

图 10 的变化趋势与图 9 中波形数据集基本相同,但精度低于图 9 波形数据集的实验结果,主要原因是森林覆盖数据集的数据概念漂移更频繁。过大的数据块影响算法的更新速度和准确率,使得算法的更新速度变慢,准确度变低。另外可以看出,在大数据块的情况下,CUE 算法的准确率比 AWE 算法的准确率低。

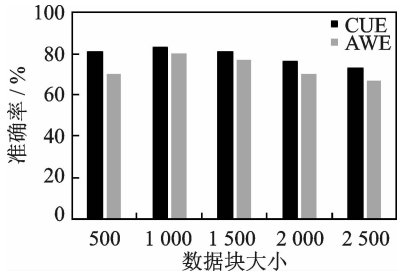

图 9 在波形数据集上数据块大小对准确率的影响对比

Fig. 9 Comparison of influence of different chunk size on accuracy on Waveform data set

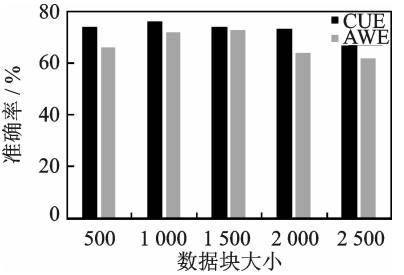

图 10 在森林覆盖数据集上数据块大小对准确率的影响对比

Fig. 10 Comparison of influence of different chunk size on accuracy on Forest CoverType data set

通过两个实验的对比可以看出,CUE 算法不用担心训练数据不足的问题并且能更快适应流的变化,因此能够提高整体分类能力,适合于较小的数据块环境。

2 基于聚类的动态选择分类器 DCSC

2.1 算法描述

DCSC 算法是基于聚类技术的数据流分类算法。该算法的提出采用动态选择思想,可以分为:(1) 基分类器的训练;(2) 采用训练好的分类器对数据流进行分类;(3) 剪枝表现比较差的基分类器,训练新的基分类器来替换差的基分类器。

2.1.1 集成分类模型的构造

首先将数据流划分成大小相等的数据块 S_1, S_2,…,S_n,按照到达顺序聚类每个数据块,把训练数据分成各不相交的簇,保留每个簇的概要信息。簇的概要信息公式为

$$\mathrm{INFO}_{\mathrm{cluster}} = < n, \mathrm{centroid} > \tag{4}$$

式中:n 为簇中训练数据的个数,centroid 为簇的聚类中心。

聚类结束后,使用 VFDT 在数据块上的训练决策树作为基分类器,并将训练好的基分类器添加到集成模型中,最多能够容纳的基分类器的个数为 L。循环该过程 $L-1$ 次,当集成模型存在 L 个基分类器时结束循环,构造集成分类器完成。集成模型可表示为:Ensemble $= (C_1, C_2, \cdots, C_L)$。与此同时,保存 DCSC 处理一个数据块后基分类器 C_i 被选择的次数 $\mathrm{CHOSEN\_NUM}_i (i = 1, 2, \cdots, L)$。

2.1.2 使用集成分类模型来进行分类

用 DCSC 算法对数据流进行分类由两部分组成:首先采用欧几里德距离计算每个基分类器对应簇与实例 R 之间的距离。欧几里德距离为

$$d(i,j) = \sqrt{(r_{i1} - r_{j1})^2 + (r_{i2} - r_{j2})^2 + \cdots + (r_{in} - r_{jn})^2} \tag{5}$$

其中 $R_i = (r_{i1}, r_{i2}, \cdots, r_{in})$ 代表实例 R_i 的属性值,$R_j = (r_{j1}, r_{j2}, \cdots, r_{jn})$ 代表实例 R_j 的属性值。根据式(5),找到数据块中最接近实例 R 的 k 个簇,并计算这些 k 个距离的总和,标记为 $\mathrm{DISTANCE}_K L$,然后计

算 k 个簇中所包含的数据量总量 Num。由于距离总和的大小象征该数据块与待分类实例的相似程度，数据量总和的大小象征在数据块上与待分类实例类似的的数据的数量的大小，所以找到满足最大 Num 值且最小 $\mathrm{DISTANCE}_k$ 的数据块所对应的基分类器。求出 Num 和 $\mathrm{DISTANCE}_k$ 的商，并将其标记为 p，如式(6)所示。

$$p=\frac{\mathrm{Num}}{\mathrm{DISTANCE}_k} \tag{6}$$

把 L 个基分类器具有最大 p 值的基分类器 C' 的分类结果作为集成模型的分类结果。在数据块中找最近 k 个簇的过程描述如图 11 所示。具体处理流程如算法 3 所示。其中，distance (centroidi, record) 表示簇质心与实例 record 之间的距离。choose_min_k(disti)是为了找出与实例 record 距离最近的 k 个质心，计算 p 值并将其作为选择分类器的依据。

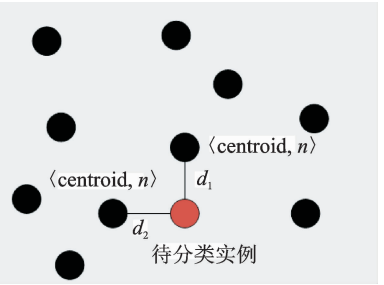

图 11 数据块中找最近的簇的示意图 ($k=2$)

Fig. 11 Diagram of finding the closest cluster in chunk($k=2$)

算法 3 使用 DCSC 对实例分类

输入：INFO cluster (簇的信息)
Ensemble (基分类器集合)
Record(数据流上到来的待分类的实例)

输出：label (分类器对 record 分类的类标)

算法描述：

```
for each INFO cluster
    for each centroid_i do
        dist_i = distance(centroid_i, record)
    end for
    choose_min_k(dist_i)
    calculate p
end for
choose the classifier with max p
label ←classify record using chosen classifier
return label
```

2.1.3 集成分类模型更新

集成分类模型的更新是为了适应数据流环境下发生的概念漂移和围标分布变化的问题，去除表现不佳的基分类器，使用最近数据训练新的基分类器加入到集成模型中。

算法的更新频率影响算法的效率。为了平衡算法的效率和更新频率，DCSC 算法在每处理一个数据块之后，通过分类准确率 accuracy 和分类数据块时分类器被选择的次数 used_times 来去除表现不佳的基分类器。MIN_FREQUENCY 表示被选择的最小次数，MIN_ACCURACY 表示最小准确率。如果在数据块的处理之后选择分类器的次数低于 MIN_FREQUENCY，则表明分类器对整个集成模型的贡献很小，与当前概念有较大差异，应该将其移除。如果选择分类器的次数大于 MIN_FREQUENCY，准确率低于 LOW_ACCURACY，表明该分类器不适应当前的概念并且应当移除该分类器。最后，通过训练新基分类器，并将训练的新基分类器添加到集成模型中完成集成模型的更新操作，完整描述如算法 4 所示。

算法 4 更新集成分类模型

输入：Ensemble (包含 L 个基分类器的集成模型)
S_n(数据流中第 n 个数据块)

输出：Ensemble (更新后的集成模型)

算法描述:

```
classify current chunk S_n
for each classifier C_i in the Ensemble
    if C_i.used_times < MIN_FREQUENCY
        then remove C_i
    else if C_i.accuracy < MIN_ACCURACY
            then remove C_i
        end if
    end if
end for
if account(Ensemble)<L then
    cluster current chunk and train C' on the current chunk S_n
    add C' to Ensemble
end if
return Ensemble
```

2.1.4 集成分类模型的整体描述

DCSC 算法将数据流划分为等大的数据块,首先对数据块进行聚类,然后训练分类器。分类器学到的规则越详细,这些实例的分类准确度就越高。选择基分类器之后,使用该分类器对实例进行分类,并将结果用作实例的最终类标签。完整描述如算法 5 所示。

算法 5 DCSC 算法

输入:S(一个数据流的实例)

输出:C(数据流上每个实例的预测类别)

算法描述:

```
数据流划分为相同大小的数据块
按顺序 L 次在每个数据块中训练一个分类器
for each chunk
    classify each instance in the chunk by Algorithm 3
    for each classifier C_i in the Ensemble
      if C_i.used_times < MIN_FREQUENCY or C_i.accuracy < MIN_ACCURACY
          update the ensemble using Algorithm 4
        end if
    end for
end for
return C
```

2.2 实验结果分析

DCSC 算法从集成模型中的基分类器中选择与待分类的实例具有最高相似度的基分类器,该实例的类标由所选择的基分类器的分类结果确定。与整合分类模型相比,DCSC 具有较快分类未知类别实例的优点,分类准确率高于单分类器。

2.2.1 数据集

本实验所使用数据流环境用 KDD CUP'99[15] 网络入侵数据集和 SEA 数据集来模拟。

(1) KDD CUP'99[15] 网络入侵数据集。该数据集是真实数据集,是麻省理工学院林肯实验室管理的局域网的两个星期内 TCP 的连接记录,均为正常连接或入侵或攻击记录。

(2) SEA 数据集。该数据集是合成数据集,在数据挖掘中经常用到,数据集中的每条记录都有 2 个类标签和 3 个连续的属性。它首先出现在文献[16]中,并被用于算法 SEA 的实验部分。

2.2.2 实验分析

两个算法的参数设置如表 2 所示。

(1) 参数影响

在 DCSC 算法中,对待分类的实例进行分类需要选择合适的基分类器,在此过程中,考虑簇中包含的数据量,找到最接近待分类实例的 k 个簇的质心。由于实验中将每个数据块上的簇数设置为 10,与待分类实例最近的簇 k 不能太大。在本节中,通过实验研究 DCSC 在 $k=1,2$ 和 3 时的精确率。

图 12 显示不同 k 值对准确率的影响。如图 12 所示,因为 $k=2$ 不仅可以保证相似性,而且兼顾到足够多的训练数据,DCSC 算法在 $k=2$ 时具有最高的准确率,因此在其他实验中,均取 $k=2$。

表 2 默认实验参数设置

Tab. 2 Default parameter setting

参数	含义	设定值
Base_num	基分类器个数	15
Chunk_size	数据块大小	1 000
Recordpoint	观测点	5 000
Tree_height	决策树的高度	8
Grace_period	分裂一个节点需要的最少的数据量	200
MIN_FREQUENCY	基分类器最少被选中的次数	50
MIN_ACCURACY	基分类器的最低准确率/%	70
MAX_REPEAT	K-means 最多迭代的次数	100

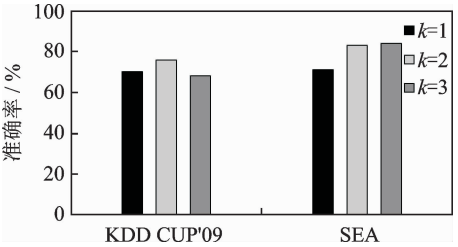

图 12 不同 k 值对准确率的影响

Fig. 12 Influence of different k on accuracy

(2) 时间效率

图 13 和图 14 显示 AWE 算法和 DCSC 算法分别在 SEA 数据集和 KDD CUP'99 数据集上的平均训练时间。如图 13,14 所示,DCSC 算法的平均训练时间与 AWE 算法相比,时而长,时而短。这是因为 DCSC 算法先进行聚类分析,并存储聚类结果,再对数据块上的基分类器进行训练,因此 DCSC 算法比 AWE 算法需要更多的时间。但是在更新频率上 DCSC 算法比 AWE 算法低。DCSC 算法只有在选择基分类器次数和准确率低于分别设定的 MIN_FREQUENCY 和 MIN_ACCURACY 值时,才会更新。然

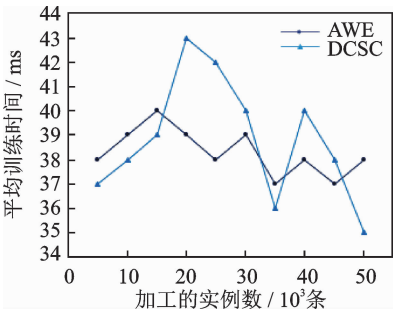

图 13 在 SEA 数据集上每个数据块的平均训练时间对比

Fig. 13 Comparison of average chunk training time on SEA data set

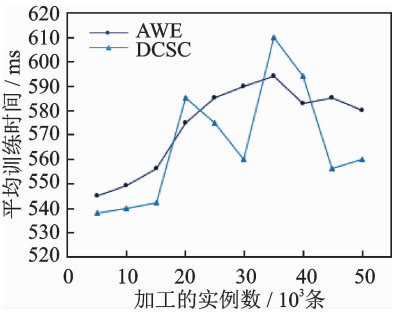

图 14 在 KDD CUP'99 数据集上每个数据块的平均训练时间对比

Fig. 14 Comparison of average chunk training time on KDD CUP'99 data set

而,AWE 算法随着新数据块的到达,都要进行决策树的训练,并将其作为后备的基分类器以供使用。除此之外,AWE 算法需要为每个数据块重新分配权重。因此,当处于平缓数据流的环境中,在更新频率上,DCSC 算法低于 AWE 算法,在平均训练时间上,DCSC 算法小于 AWE 算法。当数据流中的概念漂移或类分布发生变化时,DCSC 算法的平均训练时间将比 AWE 算法的平均训练时间长。

通过对比图 15 和图 16 可以看出,DCSC 算法在两个数据集上分类每个数据块的平均测试时间明显比 AWE 算法少。这是因为 DCSC 算法在对待分类的实例进行分类时只选择一个基分类器,这就导致没有给基分类器赋予权值的必要,所以该算法的计算复杂度减少。而在 AWE 算法中,基分类器全部都参与分类,然后综合每个基分类器分类的结果。

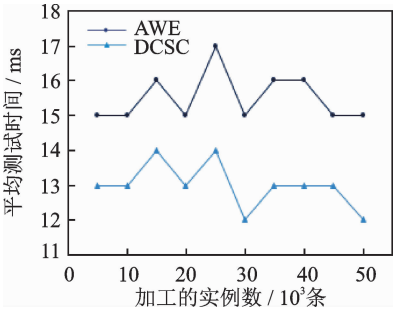

图 15 在 SEA 数据集上每个数据块的测试时间对比

Fig. 15 Comparison of average chunk test time on SEA data set

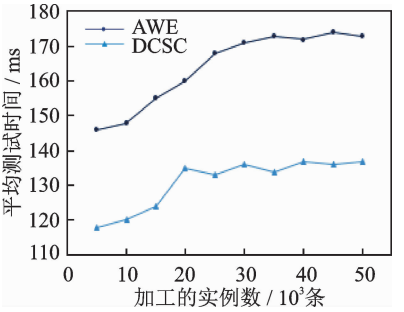

图 16 在 KDD CUP'99 数据集上每个数据块的测试时间对比

Fig. 16 Comparison of average chunk test time on KDD CUP'99 data set

(3) 内存使用情况

两种算法内存使用情况实验结果如图 17,18 所示。由图 17 和图 18 可以看出,DCSC 算法的存储占用率比 AWE 算法要高。这是因为 DCSC 需要对数据块进行聚类并保存聚类的结果信息。因此在内存占用上,DCSC 算法比 AWE 算法消耗略多。并且在聚类分析时,所消耗的内存与数据属性的数量存在正相关关系,DCSC 算法在 KDD CUP'99 数据集中消耗的内存比在 SEA 数据集上要多。

图 17 和图 18 同时也显示,算法 AWE 和算法 DCSC 随着数据流的不断流入在内存的消耗上都保持在一个恒定的范围内增长。这表明 DCSC 算法可以有效地挖掘数据流,因为在数据流挖掘中存储容量有限,DCSC 满足这一限制条件。

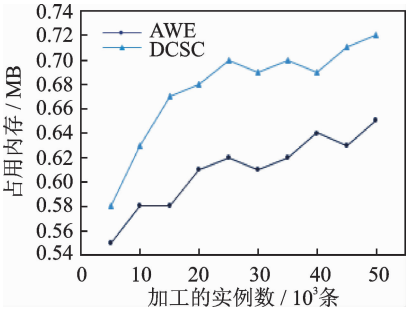

图 17 在 SEA 数据集上内存使用情况对比

Fig. 17 Comparison of memory usage on SEA data set

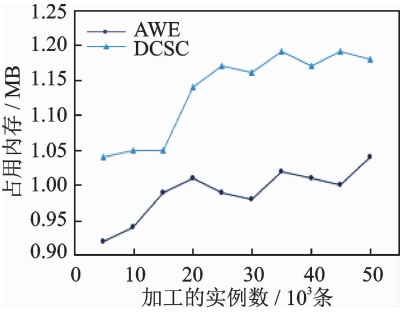

图 18 在 KDD CUP'99 数据集上内存使用情况对比

Fig. 18 Comparison of memory usage on KDD CUP'99 data set

(4) 准确率

对于 SEA 数据集,DCSC 算法的准确率始终高于 AWE 算法,如图 19 所示。这是由于 DCSC 算法构造出的模型只从集成模型中选择一个基分类器来对分类的实例进行分类,历史数据对分类不产生影响,因而准确率高。当处理 20 000 条的数据时,为了模拟突变漂移,改变 SEA 数据生成器参数。图 19 显示 AWE 算法和 DCSC 算法的准确率都快速下降并且恢复,其中 DCSC 算法的准确率恢复得更快。这是因为 DCSC 算法在对待分类的实例进行分类时选择的基分类器数量只有一个,不需要使用历史数据就可以获得分类结果,DCSC 算法能够快速适应新的概念。

在网络入侵数据集 KDD CUP'99 中,为了模拟数据流的突变漂移,在数据集上间隔较大的 3 部分分别连续抽取 10 000, 15 000, 20 000 条数据,准确率如图 20 所示。从图 20 可以看出 AWE 算法具有较高准确率。这是由于历史数据对数据流在相对平缓的环境下有较大的贡献,结合每个基分类器的结果,产生较高的准确率。DCSC 当突变漂移发生时只选择一个基分类器对分类实例进行分类,能够更快地适应概念漂移的数据流而不受到历史数据的影响。

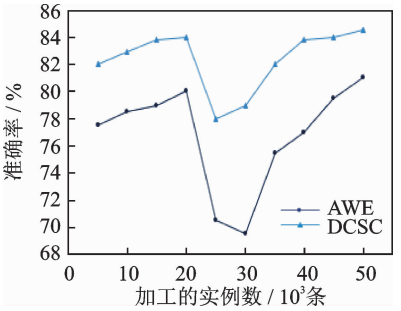

图 19 不同算法在 SEA 数据集上的准确率对比

Fig. 19 Comparison of accuracy on SEA data set

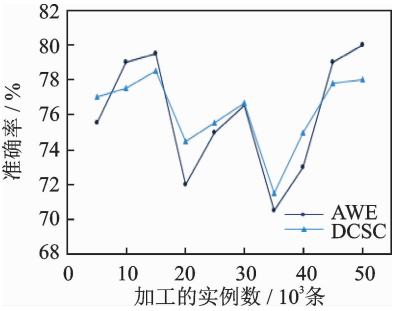

图 20 不同算法在 KDD CUP'99 数据集上的准确率对比

Fig. 20 Comparison of accuracy on KDD CUP'99 data set

3 结束语

首先,本文提出了一个动态数据流集成分类算法 CUE,使用概念自适应快速决策树 CVFDT 训练基分类器。选择概念自适应快速决策树 CVFDT 的原因,是 CVFDT 使用后续数据训练备选基分类器,更新具有较低分类准确率的基分类器。采用更新操作不仅使得基分类器之间的不相似度增加,而且使得基分类器的分类能力提高。为了使算法适应数据流漂移概念的能力得到提高,用最新的数据块更新分类准确率较低的基分类器,使基分类器适应当前的概念,使得整个集成模型的分类性能得到改善。实验表明,在具有概念漂移的数据流环境下,该算法具有快速的反应能力和良好的分类准确率。

其次,本文提出了集成分类算法 DCSC,该算法基于聚类动态选择思想。该算法首先对到来的数据流的每个数据块进行聚类,并将聚类结果保存到内存中。接下来再训练决策树,并在集成模型中加入训练好的决策树作为基分类器。集成模型保持 L 个基分类器和相应的 L 个聚类结果。在对待分类的实例进行分类时,选择与实例具有最高相似度的分类器,该分类器的分类结果作为该实例的最终分类标签。通过选择具有较低准确率且被选中次数少的基分类器,将其删除并训练新的基分类器来更新集成模型。实验表明,该算法处理具有概念漂移的数据流具有较高的时间效率,能够快速适应突发漂移。

综上所述,本文提出的对带有概念漂移的数据流进行分类的 CUE 算法和 DCSC 算法,有自己不同的优点。CUE 算法是为了增加集成模型中的基分类器之间的相异度而提出的,该算法能够实现在线更新,且对带有概念漂移的数据流,其在分类准确率和快速反应能力上都有很好的提升。DCSC 算法的提出源

于集成分类器动态选择思想,该算法对突变漂移有较快的适应能力,并且具有较高的时间效率。

参考文献:

[1] Cuzzocrea A, Gaber M M, Shiddiqi A M. Adaptive data stream mining for wireless sensor networks[M]. [S. l.]: ACM, 2014.

[2] Ko A H R, Jousselme A L, Maupin P. A novel measure for data stream anomaly detection in a bio-surveillance system[C]// Proceedings of International Conference on Information Fusion. [S. l.]: IEEE, 2011:1-7.

[3] Álvaro H, Corchado E, Sáiz J M. MOVICAB-IDS: Visual analysis of network traffic data streams for intrusion detection [J]. Lecture Notes in Computer Science, 2006, 4224:1424-1433.

[4] Widmer G, Kubat M. Learning in the presence of concept drift and hidden contexts[J]. Machine Learning, 1996, 23(1): 69-101.

[5] Wang H, Fan W, Yu P S, et al. Mining concept-drifting data streams using ensemble classifiers[C]//ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. [S. l.]: ACM, 2003:226-235.

[6] Domingos P, Hulten G. Mining high speed data streams[C]//Proceedings of the 2000 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. [S. l.]:ACM, 2000: 71-80.

[7] Papadimitriou S, Brockwell A, Faloutsos C. Adaptive hands off stream mining[C]//Proceedings of the 29th International Conference on Very Large Data Bases. Berlin, Germany: Morgan Kaufmann, 2003: 560-571.

[8] Aggarawal C, Han J W, Wang J Y, et al. On demand classification of data streams[C]//Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. [S. l.]: ACM, 2004: 503-508.

[9] Fan W. Systematic data selection to mine concept drifting data streams[C]//Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. [S. l.]: ACM, 2004: 128-137.

[10] Ding Q, Khan M, Roy A, et al. The P-tree algebra[C]//Proceedings of the 2002 ACM Symposium on Applied Computing. [S. l.]:ACM, 2002: 426-431.

[11] Zhang C S, Soda P. A double ensemble approach for classifying skewed data streams[C]//Proceeding of Advances in Knowledge Discovery and Data Mining 16th Pacific Asia Conference. Kuala Lumpur, Malaysia: Springer, 2012: 254-265.

[12] Haque A, Khan L, Baron M. SAND: Semi-supervised adaptive novel class detection and classification over data stream[C]// Proceeding of Thirtieth AAAI Conference on Artificial Intelligence. [S. l.]: AAAI, 2016.

[13] Osojnik A, Panov P, Dæroski S. Multi-label classification via multi-target regression on data streams[J]. Machine Learning, 2017, 106(6):745-770.

[14] 冯超, 文益民, 汤凌冰. 基于主要特征抽取的重现概念漂移处理算法[J]. 数据采集与处理, 2016, 31(2):315-324.
Feng Chao, Wen Yimin, Tang Lingbing. Algorithm of recurring concept drift based on main feature extraction[J]. Journal of Data Acquisition and Processing, 2016, 31(2):315-324.

[15] Bolón-Canedo V, Sánchez-Maroño N, Alonso-Betanzos A. Feature selection and classification in multiple class datasets: An application to KDD CUP 99 dataset[J]. Expert Systems with Applications, 2011, 38(5):5947-5957.

[16] Street W N, Kim Y S. A streaming ensemble algorithm (SEA) for large-scale classification[C]// ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. [S. l.]: ACM, 2001:377-382.

作者简介:

韩东红(1968-),女,副教授,硕士生导师,研究方向:数据流管理、数据挖掘,E-mail: handonghong@cse.neu.edu.cn。

马宪哲(1990-),男,硕士,研究方向:数据挖掘。

李莉莉(1993-),女,硕士,研究方向:数据挖掘、社交媒体挖掘。

王国仁(1966-),男,教授,博士生导师,研究方向:分布式数据库、生物信息学、大数据分析。

(编辑:陈珺)