一种新的数据流模糊聚类方法

孙力娟*①② 陈小东① 韩 崇① 郭 剑①②
①(南京邮电大学计算机学院 南京 210003)
②(南京邮电大学江苏省无线传感网高技术研究重点实验室 南京 210003)

摘 要：针对数据流上的聚类任务受到时间、空间限制等问题，该文提出一种基于权值衰减的数据流模糊微簇聚类算法(WDSMC)。该算法使用改进的带权值的模糊C均值算法进行处理，并采用微簇结构和权值时间衰减结构提高聚类质量。实验表明，相对于现有的数据流加权模糊 C 均值聚类(SWFCM)算法和 StreamKM++算法而言，WDSMC 算法具有更好的聚类精度。

关键词：数据挖掘；数据流；模糊C均值聚类；权值衰减；微簇聚类
中图分类号： TP301 **文献标识码**： A **文章编号**：1009-5896(2015)07-1620-06
DOI: 10.11999/JEIT141415

New Fuzzy-Clustering Algorithm for Data Stream

Sun Li-juan①② Chen Xiao-dong① Han Chong① Guo Jian①②
①(*College of Computer*, *Nanjing University of Posts and Telecommunications*, *Nanjing* 210003, *China*)
②(*Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks*, *Nanjing University of Posts and Telecommunications*, *Nanjing* 210003, *China*)

Abstract: There is a great challenge in the data stream clustering due to a limitation of time and space. In order to solve this problem, a new fuzzy-clustering algorithm, called Weight Decay Streaming Micro Clustering (WDSMC), is presented in this paper. The algorithm uses a reformed weighted Fuzzy C-Means (FCM) algorithm, and improves the quality of clustering by the structures of micro-clusters and weight-decay. Experimental results show that this algorithm has better accuracy than Stream Weight Fuzzy C-Means (SWFCM) and StreamKM++ algorithm.

Key words: Data mining; Data stream; Fuzzy C-Means (FCM); Weight decay; Micro-clustering

1 引言

随着计算机、通信技术的发展以及大数据的到来，在如传感器网络、气象分析、股票市场、计算机网络入侵检测等众多应用领域产生了一些海量、高速、动态到达的数据，这些连续到达的数据被称之为数据流。传统的数据挖掘一般是建立在数据集是有限的，由静态概率分布产生并能够在内存随机存取的基础之上。但是对于数据流而言，传统处理静态数据集上的挖掘算法受到如下条件限制[1]：(1)数据对象是连续不断到达的；(2)无法控制数据对象的处理顺序；(3)数据流的大小是没有边界的；(4)数据对象被处理完后就会被丢弃。在实际处理时，可以使用备忘机制将部分数据保存一段时间后再丢弃；(5)数据可能是动态分布的而非单一的静态分布。聚类分析在数据挖掘领域中扮演一个重要的角色。聚类是一个把数据对象划分成多个组或簇的过程，使得簇内的对象具有很高的相似性，但与其他簇中的对象很不相似[2]。

文献[3]中提出的 Brich 算法是最早提出的可以用于数据流的聚类算法，选择正确的参数对 Brich 算法实现效率很重要，需要专业领域的知识。文献[4]提出一种称为 Clustream 的数据流聚类算法。它将整个聚类过程分成了两个部分：在线部分和离线部分。在线部分通过设定好的删除和合并原则增量更新微簇的特征向量。离线部分使用改进的 K-means 算法将微簇聚类成最终的簇，通过倾斜的时间窗口可以在用户指定的不同时间粒度内发现簇。文献[5]和文献[6]所提出基于密度的数据流聚类算法，能够得到任意形状的簇结构，但不能随着数据流的变化而动态改变，也无法在多种层次粒度下对数据流进行聚类。文献[7]提出的 StreamKM++算法是一种使用树结构和桶结构的数据流聚类方法。该

2014-11-05 收到，2015-03-20 改回，2015-06-01 网络优先出版
国家自然科学基金(61171053, 61300239)，教育部博士点基金(20113223110002)，中国博士后科学基金(2014M551635)和江苏省博士后科研资助计划项目(1302085B)资助课题
*通信作者：孙力娟 sunlj@njupt.edu.cn

算法将数据点不断放入桶中，当两个桶满时，通过调用 Coreset 树将桶合并，以此类推，最后将得到的数据点使用 kmean++[8]算法合并成簇中心。

上述文献中的算法都属于是硬聚类算法，即每一个数据点不是完全属于这一个簇，就完全属于另一个簇。这种算法不具有一般意义，因为在模糊逻辑的思想中，一个数据点可能同时是许多簇的成员。文献[9]提出的模糊 C 均值(Fuzzy C-Means, FCM)算法是数据集上的模糊聚类算法，通过不断迭代计算簇中心和隶属度矩阵，求解目标函数的最小值。数据流权值模糊 C 均值[10](Stream Weight Fuzzy C-Means, SWFCM)算法将 FCM 扩展到数据流上，由于该算法每部分仅保留簇中心点及其权值，相对于原始数据信息丢失量过大，聚类的效果并不是十分理想。针对数据流在不同时间段可能形成不同聚类模式的问题，文献[11~13]提出了在数据流处理过程中可能出现的概念漂移检测和处理方法。文献[14]使用一种延迟处理的方法解决数据流聚类中的孤立点检测问题。在数据流的进化过程中，用户可能对最近到达的数据流更感兴趣，而 Brich, StreamKM++, SWFCM 等算法都没有考虑过这个问题。本文在研究了一系列聚类算法基础上，提出了一种基于权值衰减的数据流模糊微簇聚类算法(Weight Decay Streaming Micro Clustering, WDSMC)，使用微簇结构提高聚类准确性，使用权值时间衰减结构降低历史数据的影响。

2　基本概念

定义 1(数据流模型)：　数据流是一个带有时间戳的多维的数据点集合 $\boldsymbol{X}_1,\boldsymbol{X}_2,\cdots,\boldsymbol{X}_s,\cdots$。每个数据点 $\boldsymbol{X}_i$ 是一个包含 d 维的数据记录，表示为 $\boldsymbol{X}_i=(x^1,x^2,\cdots,x^d)$。

定义 2(模糊簇)：　给定一个数据集 $\boldsymbol{X}=\{\boldsymbol{X}_1,\boldsymbol{X}_2,\cdots,\boldsymbol{X}_n\}$，模糊簇 $\boldsymbol{C}_j$ 是 $\boldsymbol{X}$ 的一个子集，它允许 $\boldsymbol{X}$ 中每个数据点都具有一个属于 $\boldsymbol{C}_j$ 的 0 到 1 之间的隶属度[15]。

定义 3(隶属度矩阵)：　隶属度矩阵 $\boldsymbol{U}$ 是一个 $m\times n$ 的矩阵，其中 m 是需要聚类的簇个数，n 是数据点的个数，u_{ij} 表示第 $\boldsymbol{X}_j$ 在模糊簇 $\boldsymbol{C}_i$ 中所占的比重。

关于 u_{ij} 的取值范围，满足如下两条性质[15]：

性质 1　对于每个数据点 $\boldsymbol{X}_j$ 和簇 $\boldsymbol{C}_i$，都有 $0\le u_{ij}\le 1$。

性质 2　对于每个数据点 $\boldsymbol{X}_j$，都有 $\sum_{i=1}^{m}u_{ij}=1$。

定义 4(簇中心点)：　第 i 个模糊簇的中心点 $\boldsymbol{V}_i$ 应该结合隶属度矩阵计算，其计算公式为[9](p 的含义见定义 5)

$$\boldsymbol{V}_i=\sum_{j=1}^{n}(u_{ij})^p\boldsymbol{X}_j\Big/\sum_{j=1}^{n}(u_{ij})^p \tag{1}$$

定义 5(代价函数/目标函数)：　按照聚类 C 的误差平方和来评价聚类的质量，其公式为[9]

$$J=\sum_{i=1}^{m}\sum_{j=1}^{n}(u_{ij})^p(d_{ji})^2 \tag{2}$$

其中，加权指数 p($p\ge 1$)控制隶属度的影响。p 的值越大，隶属度的影响越大；$d_{ji}=\|\boldsymbol{X}_j-\boldsymbol{V}_i\|$ 表示数据点 $\boldsymbol{X}_j$ 与第 i 个簇的中心 $\boldsymbol{V}_i$ 的欧式空间距离。

目标函数 J 的值越小，表明聚类质量越好。要求式(2)取值最小，结合 $\sum_{i=1}^{m}u_{ij}=1$ 的特点，可以得到隶属度矩阵的计算[9]：

$$u_{ij}=1\Big/\sum_{k=1}^{m}\left(\frac{d_{ij}}{d_{kj}}\right)^{2/(p-1)} \tag{3}$$

3　基于权值衰减的数据流模糊微簇聚类 WDSMC 算法

本文提出的基于权值衰减的数据流模糊聚类 WDSMC 算法是将数据流分块处理，使用改进的带权值模糊 C 均值算法(Advanced Weighted Fuzzy Clustering Method, AWFCM)微簇聚类分块后的数据流，在每一部分聚类得到微簇中心点及其权值进入下一部分数据流之前，衰减前一部分数据点权值，减小历史数据对于最近阶段聚类结果的影响。在处理完整个数据流之后，将得到微簇中心及其权值当做数据点，再次调用 AWFCM 算法处理，最终得到整个数据流上的各个簇中心。

3.1　模糊 C 均值聚类(FCM)算法及其改进

FCM[9]是一种模糊聚类算法，其算法步骤如表 1 所示。

从表 1 可以看出，FCM 算法的初始隶属度函数是随机选择的，只要满足性质 2 即可，即所计算出的初始簇中心也是随机的，与样本数据点无关。因 FCM 算法是局部收敛的，所以 FCM 的算法性能强

表 1　FCM 算法

步骤 1　初始化 $m\times n$ 的隶属度矩阵 $\boldsymbol{U}$，其中 m 是需要聚类的簇个数，n 是数据点个数，u_{ij} 的值随机从(0,1)中选择且满足性质 2；
步骤 2　根据式(1)计算每一个模糊簇的中心点 $\boldsymbol{V}_i$，其中 i=1,2,$\cdots$,m；
步骤 3　根据式(2)计算目标函数 J，如果目标函数值达到最小即满足下面 3 种情况之一：(1)其值收敛于某一个固定的值；(2)其值前后两次迭代的差小于某一个指定的阈值；(3)迭代次数达到某一个指定的值，则停止迭代，否则向下继续执行；
步骤 4　根据式(3)计算新的隶属度矩阵 $\boldsymbol{U}$，返回步骤 2。

烈依赖于初始簇中心的选择。本文针对上述 FCM 算法的缺陷，提出了改进 FCM 算法(AFCM)，如表 2 所示。

表 2 改进的 FCM 算法(AFCM)

步骤 1 在样本数据中随机选择一个数据点作为第 1 个簇的中心点；
步骤 2 将剩余的每个点与上一个点的欧式空间距离按大小排列，然后按照轮盘的方式选取一个数据点作为下一个簇的中心点。即距离越大，在轮盘上所占的面积越大，被选中的概率越大；
步骤 3 更新剩余每个点到其最近簇中心的距离，继续按照上述方法选择下一个簇的中心点；
步骤 4 重复步骤 3，直到 m 个簇的中心点都被选取出来；
步骤 5 选取出来的初始中心点，根据式(3)计算隶属度矩阵；
步骤 6 从表 1 的步骤 2 开始调用，进行余下的 FCM 算法的处理过程。

从表 2 中可以看出，AFCM 选择距离比较远的数据点作为初始簇中心，这符合簇内对象具有很高相似性，而簇间对象相似性低的要求。降低算法陷入局部收敛的概率，提高聚类质量。为了不陷入局部最优解中，WDSMC 算法多次独立调用 AFCM 算法处理数据流的第 1 部分，选取使得目标函数值最小的微簇中心及其权值进入数据流的下一部分。而对于数据流的其余部分，上一部分聚类得到微簇中心点就作为下一部分的初始微簇中心点，从而加快收敛速度。

3.2 权值时间衰减与微簇结构

数据流是按照时间顺序依次到达的数据点。随着时间的推进，用户往往更加在意最近到达的数据点所形成的簇。本文提出的 WDSMC 算法采用权值衰减结构提高最终所形成簇的时效性。

定义 6(数据点权值) 在 t 时刻，数据点 j 的权值 w 定义为

$$w_j^t = 2^{-\lambda(t-t_d)}, \quad 1 \le j \le n \tag{4}$$

其中 λ 是衰减因子，t_d 是数据点到达的时刻。最近到达的数据点的权值为 1，上一部分的数据点权值为 $2^{-\lambda}$，依次类推。通过给每一个数据附加一个随时间衰减的权值，降低历史数据的影响。由式(4)可见，λ 的取值越大，旧的数据衰减的越快，对最终簇的形成影响越小。

定义 7(簇权值) 在 t 时刻，第 i 个簇的权值 cw 定义为

$$\mathrm{cw}_i^t = \sum_{j=1}^{n} u_{ij} w_j^t, \quad 1 \le i \le m \tag{5}$$

即所有数据点在该簇上的隶属度与权值乘积之和表示该簇的权值。每个簇的权值之所以说是随时间衰减，是因为簇中的数据点是随时间衰减的，由式(4)和式(5)，可以得出的结论为：

性质 3 从 t 时刻到 $t+1$ 时刻，第 i 个簇权值 cw 将衰减 $2^{-\lambda}$，即满足式(6)：

$$\mathrm{cw}_i^{t+1} = \mathrm{cw}_i^t \times 2^{-\lambda}, \quad t = 0,1,2,\cdots \tag{6}$$

在数据流分块处理过程中，如果仅仅把 m 个簇的中心点和权值传递到下一部分的数据流中，由于包含的原始数据点信息太少，影响到聚类质量。在 WDSMC 算法中，每部分按更大的簇个数 k 进行聚类，称之为微簇。使用微簇可以保留更多的簇中心点和权值，最后使用 AWFCM 算法将微簇聚类成最终的簇。

3.3 带权值的 FCM 算法(WFCM)

因为在 WDSMC 算法中将每个数据点都赋了一个随时间衰减的权值，所以在使用 FCM 算法处理每一部分的数据时必须要考虑到权值的因素。WFCM 与 FCM 的区别在于式(1)和式(3)要加上权值的影响：

式(1)需要变换为式(7)：

$$\boldsymbol{V}_i = \sum_{j=1}^{n} w_j (u_{ij})^p \boldsymbol{X}_j \Big/ \sum_{j=1}^{n} (u_{ij})^p \tag{7}$$

式(2)需要变换为式(8)：

$$J = \sum_{i=1}^{m}\sum_{j=1}^{n} w_j (u_{ij})^p (d_{ji})^2 \tag{8}$$

3.4 WDSMC 算法框架

本节给出 WDSMC 数据流聚类算法框架，如图 1 所示，依据数据流到达时间的次序将数据流分成形如 $S_1,S_2,\cdots,S_p$ 的块，数据块的大小由主机内存的大小决定，记数据块 $S_1,S_2,\cdots,S_p$ 中数据点的个数分别为 $n_1,n_2,\cdots,n_p$。

每一部分处理过程如下：新到达的数据点的权

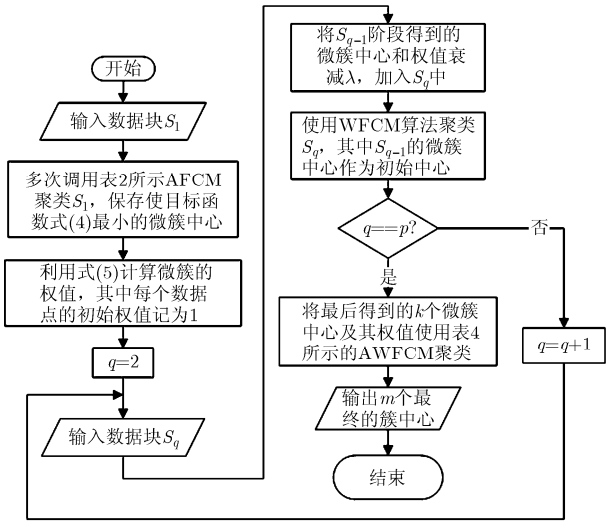


图 1 WDSMC 算法流程图

值记为 1，上一部分计算得到的微簇中心及其权值也当作数据点处理，并使用式(6)对上一部分微簇权值进行衰减处理；依据式(3)计算隶属度，利用式(7)和式(5)计算更新微簇中心及其权值；通过不断地的迭代，使得式(8)达到最小或者到达到某一指定迭代次数。WDSMC 算法框架伪代码如表 3 所示。

在表 3 中数据块 $S_q(q>1)$的处理过程中，是将 S_{q-1} 中的微簇中心当作一般的数据点，与 S_q 中的新的数据点一起聚类。也就是说，算法中所用到的各个方程式中，其数据点个数 n 是 $S_q(q>1)$中数据点个数与 S_{q-1} 中微簇个数之和，即满足：$n=n_q+k$。如何通过这 k 个微簇中心点及其权值得到最终的 m 个用户需要的簇呢？本文使用表 4 所示的带权值模糊 C 均值算法 AWFCM 来完成该任务。

表 3　WDSMC 算法框架

步骤 1　输入数据块 $S_q(1 \le q \le p)$；

步骤 2　更新微簇中心权值：

(1)若 $q=1$：　多次独立调用 AFCM 算法，取使目标函数式(8)最小的数据点微簇中心 $v_i, i=1,2,\cdots,k$，k 是微簇的个数，根据式(5)计算微簇中心权值，其中最近到达每个数据点的权值为 1，转至步骤 7；

(2)若 $q>1$：

(a)将数据块 S_{q-1} 聚类得到的各个微簇的权值依据式(6)衰减 $2^{-\lambda}$；

(b)将数据块 S_{q-1} 的微簇中心和衰减后的权值当作普通的数据点，与 S_q 中的数据点一起聚类，S_q 中的新到达的数据点权值记为 1；

(c)将 S_{q-1} 中的微簇中心点作为对 S_q 聚类时的初始中心，依据式(3)计算隶属度；

步骤 3　根据式(5)计算微簇中心点权值；

步骤 4　依据式(7)更新微簇中心；

步骤 5　依据式(8)计算目标函数，若目标函数值达到最小即满足下面 3 种情况之一：(1)其值收敛于某一个固定的值；(2)其值前后两次迭代的差小于某一个指定的阈值；(3)迭代次数达到某一个指定的值，则转至步骤 7，否则执行步骤 6；

步骤 6　依据式(3)更新隶属度矩阵 $\boldsymbol{U}$，返回步骤 3；

步骤 7　如果 $q == p$，则结束，否则转至步骤 1。

表 4　AWFCM 算法

输入：k 个微簇中心点及其权值

输出：m 个最终簇中心

步骤 1　使用改进 FCM 选取初始中心点的方法，在 k 微簇中心点中选择 m 个点作为初始中心点；

步骤 2　使用式(3)计算隶属度矩阵 $\boldsymbol{U}$；

步骤 3　使用式(7)计算簇中心 $v_i, i=1,2,\cdots,m$；

步骤 4　使用式(8)计算目标函数 J，当 J 值收敛某一值或者达到迭代次数时，停止并输出所得到的 m 个最终簇中心；否则返回步骤 2。

4　实验结果

本文使用 Matlab 实现了 WDSCM 算法，同时实现了用来做比较分析的 SWFCM[10]算法(Matlab 实现)和 StreamKM++[7]算法(Microsoft Visual C++实现)。所有的实验都是在一个 2G 内存、酷睿 2 双核 CPU、操作系统为 Windows 7 的主机上运行的。SWFCM 算法和 StreamKM++算是两种常见的数据流聚类算法，其中 SWFCM 属于模糊聚类算法，而 StreamKM++属于硬聚类算法。所有实验的聚类质量评价标准是依据式(2)得到的，即由数据点与簇中心的距离的平方与隶属度乘积之和所决定。

4.1　数据集

本文采用来自真实世界的数据源，以期获得具有实际指导意义的结果。所使用的数据主要来自 UCI Machine Learning Repository[16]。表 5 总结了本文实验即将用到的 2 个数据集(数据维度不包括类标号)：

表 5　实验使用的数据集

数据集	数据点个数	数据维度	数据类型
Spambase	4601	57	浮点型
Covertype	311079	34	整形，浮点型

4.2　算法参数

本小节主要介绍 WDSCM 以及将要做对比的 SWFCM, StreamKM++算法的一些基本的参数设置。StreamKM++所有参数与文献[9]中介绍的一样。SWFCM 算法和 WDSCM 算法在使用 FCM 或其改进时，最大迭代次数都为 50 次；计算目标函数的加权指数 $p=3$；目标函数达到收敛条件满足其前后值差的绝对值小于 10^{-6}。若非特别说明，WDSCM 算法的权值时间衰减因子设为 $\lambda=0.125$，微簇个数设为 $k=400$。在从微簇得到最终簇的过程中，采用 5 次独立调用改进的带权值的 FCM 算法。在处理数据流第 1 部分时也是采用 5 次独立调用改进的 FCM，降低第 1 部分初始微簇中心随机选择带来的误差。

4.3　λ 的取值分析

权值时间衰减因子 λ 是 WDSMC 算法中很重要的一个参数，它决定了历史数据对当前簇的影响程度。在本文的其他实验中，λ 取一个较小的值 0.125，而对于那些需要在最近的数据流上聚类的应用中，可以适当增大 λ 的值，算法具有很好的灵活性。

本文取 λ 的值从 0.25 到 16，评估算法在 Spambase 数据流上的聚类效果，其中数据流的每一部分有 1000 个数据点，最后一部分有 601 个数据点，最终聚类簇个数 $m=4$，微簇个数 $k=80$，其余参数设置同上一节所介绍，实验结果图 2 所示。

在图 2 中实线段代表最终簇中心在整个 Spambase 数据集上的目标函数值，虚线段代表

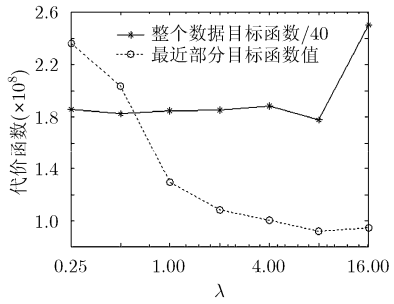


图 2 λ取值与代价函数的关系

最终簇中心在 Spambase 最后一部分上的目标函数值。随着λ取值的增加，算法在最近部分的数据上的代价函数越来越小，最近部分的聚类效果越来越好；而在整体数据流上计算的代价函数值不会因为λ在一定范围内增加而有明显的变大，即整体聚类质量不会因为λ的增大而大幅降低。这是本文提出算法的优势所在，提供了一定的灵活性，可以依据具体的应用需求选择合适的λ值。

4.4 算法性能对比

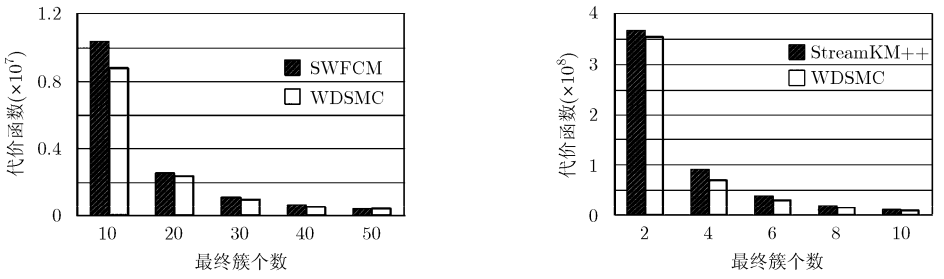
由于是在整个数据集上计算代价函数，为公平起见，在与 SWFCM 和 StreamKM++算法做性能对比时，本文算法暂不考虑权值时间衰减函数的影响，即$\lambda = 0$。每种算法下的聚类代价函数如图 3 和图 4 所示，由于 StreamKM++算法是一种硬聚类方法，其模糊聚类下的代价函数是通过将其聚类后的簇中心和原数据集代入式(3)和式(2)所得到的。实验发现，虽然 SWFCM 算法与 StreamKM++算法和 WDSMC 算法相比在运行时间上是较好的，但是其聚类质量(代价函数越小，聚类质量越高)却不如另外两种算法，特别是随着最终簇个数的增加，与后两种算法之间的差距更加明显。

与 SWFCM 和 StreamKM++算法相比，本文算法还有如下两点优势：(1)由于本文算法在第 1 步使用微簇聚类，不需要用户事先指定最终簇的个数。因此可以像 Clustream 算法一样，将聚类过程分成在线和离线两个部分，只要保留微簇中心及其权值，就可适应聚类成不同最终簇个数的需求。使用微簇即提高了聚类的精确性，又减少了在修改簇个数时需要重新聚类的麻烦。(2)本文算法可以根据需要，适当增大λ值，减小历史数据对聚类的影响，适用于实时性要求高的业务。

4.5 微簇聚类最终簇

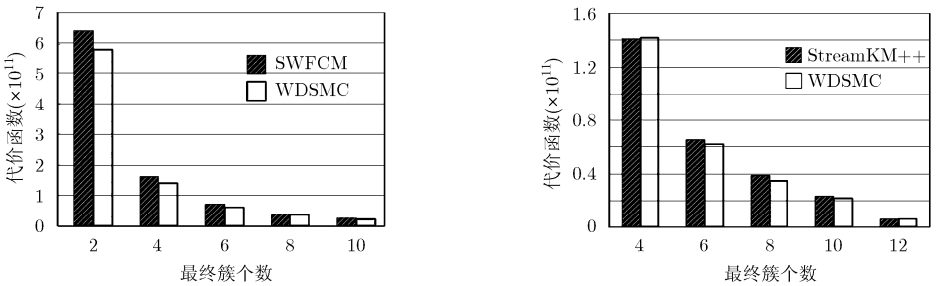
在本文前面的实验中，将数据流挖掘得到的微簇二次聚类成最终簇时使用的是表 4 所示的 AWFCM 算法。本节考虑一种不同的微簇聚类成最终簇的方式 —— 遗传模拟退火算法(Simulated Annealing Genetic Algorithm, SAGA)[17]。该算法将遗传算法和模拟退火算法相结合用于聚类分析，是一种非线性的全局最优化搜索方法，通过不断尝试跳出局部最优解的途径得到优化问题的全局最优解，相对而言 FCM 算法以及 AFCM 算法都是一种局部最优搜索方法，故使用该算法对比本文所提出的 AFCM 算法的聚类效果。

表 6 列出了 AWFCM 算法与 SAGA 算法将 Spambase 数据集上的微簇聚类成最终簇所用的时间和聚类目标函数对比，其中 SAGA 算法中的参数设置与文献[17]相同。作为一种非线性优化方



(a) SWFCM与WDSMC算法的性能对比 (b) StreamKM++和WDSMC算法的性能对比

图 3 Spambase 数据集上实验结果



(a) SWFCM与WDSMC算法的性能对比 (b) StreamKM++和WDSMC算法的性能对比

图 4 Covertype 数据集上的实验结果

法，模拟遗传退火算法 SAGA 通过大量的迭代可以达到全局最优的搜索效果，但是需要消耗大量的时间。在表 6 中，对于实验中 Spambese 数据集从微簇聚类到最终簇的过程，使用 SAGA 算法需要的时间是 AWFCM 算法的 300～400 倍，而对聚类质量的提升(对应于代价函数的减小)不足 1%。实验证明，改进的带权值的 FCM 算法在保证聚类质量的前提下，大大提升了算法的时间效率。

表 6 在 Spambase 数据集上运行时间和代价函数对比

簇个数 m	运行时间(s)		代价函数	
	AWFCM	SAGA	AWFCM	SAGA
2	2.342	677.716	3.56×10^8	3.56×10^8
4	2.881	1584.842	7.34×10^8	7.34×10^7
6	4.120	1630.182	2.94×10^7	2.92×10^7
8	4.264	1392.401	1.43×10^7	1.42×10^7
10	5.375	1848.038	8.67×10^7	8.66×10^7

5 结束语

本文提出了一种新的数据流模糊聚类算法。该算法使用微簇结构保存最终聚类所需要的信息，提高聚类质量和自适应最终簇的个数；使用权值时间衰减结构降低历史数据对聚类的影响。通过对不同数据集进行实验表明，本文所提出的 WDSCM 算法相对于 StreamKM++和 SWFCM 算法而言具有更好的整体聚类效果。在微簇聚类最终簇的过程中分别使用遗传模拟退火算法 SAGA 和改进的带权值的 FCM 算法对比算法性能。接下来的工作方向可以包括以下主题：探究每部分数据块大小变化对最终聚类效果的影响；改进微簇结构，以进一步提高聚类质量；关注不同时间段数据流簇的演化情况和在算法加入噪声点的检测处理过程等。

参 考 文 献

[1] Jonathan A S, Elaine R F, Rodrigo C B, *et al.*. Data stream clustering: a survey[J]. *ACM Computing Surveys*, 2013, 46(1): 13:1–13:31.

[2] Shifei D, Fulin W, Jun Q, *et al.*. Research on data stream clustering algorithms[J]. *Artificial Intelligence Review*, 2013, 43(4): 593–600.

[3] Tian Z, Raghu R, and Miron L. BIRCH: an efficient data clustering method for very large databases[C]. Proceedings of the ACM SIGMOD International Conference on Management of Data, New York, USA, 1996: 103–114.

[4] Aggarwal C C, Han J, and Yu P S. A framework for clustering evolving data streams[C]. Proceedings of the 29th Conference on Very Large Data Bases, Berlin, Germany, 2003: 81–92.

[5] Chen Y and Tu L. Density-based clustering for real-time stream data[C]. Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, USA, 2007: 133–142.

[6] Cao F, Ester M, Qian W, *et al.*. Density-based clustering over an evolving data stream with noise[C]. Proceedings of the 16th SIAM International Conference on Data Mining, Maryland, USA, 2006: 328–339.

[7] Ackermann M R, Märtens M, Raupach C, *et al.*. StreamKM ++: a clustering algorithm for data streams[J]. *Journal of Experimental Algorithmics*, 2012, 17(1): 2–4.

[8] Arthur D and Vassilvitskii S. K-means++: the advantages of careful seeding[C]. Proceedings of the 2007 ACM-SIAM Symposium on Discrete Algorithm, New Orleans, USA, 2007: 1027–1035.

[9] Baraldi A and Blonda P. A survey of fuzzy clustering algorithms for pattern recognition[J]. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 1999, 29(6): 778–785.

[10] Renxia W, Xiaoya Y, and Xiaoke S. A weighted fuzzy clustering algorithm for data stream[C]. Proceedings of the 2008 ISECS International Colloquium on Computing, Communication, Control, and Management, Guangzhou, China, 2008: 360–364.

[11] 郭躬德, 李南, 陈黎飞. 一种基于混合模型的数据流概念漂移检测算法[J]. 计算机研究与发展, 2014, 51(4): 731–742.
Guo Gong-de, Li Nan, and Chen Li-fei. Concept drift detection for data stream based on mixture model[J]. *Journal of Computer Research and Development*, 2014, 51(4): 731–742.

[12] 胡伟. 一种改进的动态 k-均值聚类算法[J]. 计算机系统应用, 2013, 22(5): 116–121.
Hu Wei. Research and realization of a web information extraction and knowledge presentation system[J]. *Application of Computer System*, 2013, 22(5): 116–121.

[13] 李子柳. 大数据实时流式聚类框架研究[D]. [硕士论文], 中山大学, 2013.
Li Zi-liu. A framework for real time stream clustering of big data[D]. [Master dissertation], Sun Yat-sen University, 2013.

[14] Hossein M K, Suhaimi I, and Javad H. Outlier detection in stream data by clustering method[J]. *International Journal of Advanced Computer Science and Information Technology*, 2013, 2(3): 25–34.

[15] Jiawei H, Micheline K, Jian P. 范明, 孟小峰. 数据挖掘: 概念与技术[M]. 第 3 版, 北京: 机械工业出版社, 2012: 323–350.

[16] David Aha. UCI Machine Learning Repository[OL]. https://archive.ics.uci.edu/ml, 2014.

[17] 史峰, 王辉, 郁磊, 等. Matlab 智能算法: 30 个案例分析[M]. 北京: 北京航天航空大学出版社, 2011: 188–196.

孙力娟：女，1963年生，博士，教授，博士生导师，研究方向为演化计算、无线传感器网络和数据处理等.

陈小东：男，1992年生，硕士生，研究方向为数据挖掘、数据处理.

韩　崇：男，1985年生，博士，讲师，研究方向为多媒体信息处理、数据挖掘.

郭　剑：男，1978 年生，博士，副教授，硕士生导师，研究方向为无线传感器网络、无线多媒体传感器网络.