一种基于高斯混合模型的不确定数据流聚类方法

曹振丽 [1,2]　孙瑞志 [1]　李勐 [1]

[1] (农业部农业信息获取技术重点实验室 北京 100083)

[2](中国农业大学烟台研究院 烟台 264670)

(E-mail: caozhenli2004@163.com)

A Method for Clustering Uncertain Data Streams Based on GMM

Cao Zhenli [1,2], Sun Ruizhi [1], Li Meng[1]

[1](Key laboratory of Agricultural information acquisition technology, China Agricultural University, Beijing 100083)

[2](Yantai Academy, China Agriculture University, Yantai 264670)

Abstract With the sensors widely used, it brings a lot of uncertain data streams. When the input datas are continuously random variables, the existing clustering method based on discrete random variables can not meet the requirements of efficiency and accuracy. In order to solve the problem mentioned above , we propose a new method which was named cmicro algorithm. First, we use the Gaussian mixture model as the basic representation of uncertain data streams. Second, we propose a clustering method which can find clustering in time dimension. This method can make up for the deficiency of traditional clustering which can't find the non-spherical clustering. Third, we discuss the influence of the different parameter values by experiment. Finally, the compared result shows that the proposed algorithm promotes the accuracy of clustering.

Key words Gaussian Mixture Model; uncertain data streams; clustering; sensor; sketch

摘　要　传感器的广泛应用产生了大量的不确定数据流，在聚类应用中，当输入数据为连续型随机变量时，现有基于离散型随机变量的聚类方法无法满足数据流应用在效率和精度上的要求。本文使用高斯混合模型作为不确定数据的基本表示形式，仅需要保存不同组件的描述信息即可，可以更好的利用存储空间，完成对真实情况的逼近，并提出了一种可以发现时间维度上的不确定数据流聚类方法 cumicro, 该算法将时间直接作为数据属性，可直接查询某个时间维度的聚簇，避免了传统基于划分的聚类中较难发现非球状聚簇的问题。通过实验与经典算法 umicro 进行比较，证明了本文算法的有效性，并分析了不同 K 值、Tau 值下的聚类结果。最后得出结论，原始数据较密集时，相较原有基于离散模型的聚类，该算法具有准确度上的优势。

关键词　高斯混合模型;不确定数据流;聚类;传感器; 概要结构

中图法分类号　**TP393**

收稿日期: yyyy-mm-dd

基金项目:国家科技支撑计划（2012BAH26B01）、山东省科技发展计划（2014GNC110026）.

作者简介：曹振丽（1979-），中国农业大学信息与电气工程学院博士生，中国农业大学烟台研究院讲师，主要从事农业信息化、云计算与大数据等方面的研究. 通讯作者：孙瑞志，教授，博士生导师，主要从事农业信息化技术、云计算与大数据、计算机支持的协同工作方面的研究，E-mail: sunrz_cn@sina.com.cn

1 引言

随着物联网技术的发展，利用传感器采集到的大数据成为人们进行事件分析和决策的重要依据，但由于采集设备的误差和故障等因素的存在，数据的可获取性、准确性和实时性都会受到影响[1]。针对这些具有概率特征的不确定的数据，人们采用了不确定数据表示方法来进行统一描述。不确定数据流是不确定数据的一种主要表现形式，是不确定数据的有序序列。对不确定数据流进行聚类分析，在环境监测、实时监控、网络入侵检测等应用中被广泛使用，具有较高的实用价值，但目前主要的不确定数据流聚类方法主要存在两方面的问题，首先，对于连续型随机变量的输入,现有方法一般通过“抽样——直方图”的方法来实现，这样会造成精度损失；其次，目前的不确定数据流聚类在概要结构设计和时间的演化分析上并不完善，较难发现时间维度上的聚簇。本文使用高斯混合模型表示数据的不确定性，设计了一种聚类算法以弥补现有算法在以上两方面的不足。

本文提出了一种针对不确定数据流的聚类算法 cumicro，主要工作包括:1) 充分考虑存储空间的影响，提出了使用高斯混合模型描述原始的不确定数据的不确定性；2) cumicro 算法通过对时间属性化处理，弥补了目前不确定数据流聚类在概要结构设计和时间的演化分析上并不完善，较难发现时间维度上的聚簇的不足；3) 设计了一种基于高斯混合模型表示的不确定数据流聚类算法 cumicro。最后,通过真实数据集的实验表明,cumicro 具有良好的聚类质量,能够有效适应不确定数据流场景。

本文第 1 节介绍相关工作。第 2 节设计了不确定数据流聚类算法框架,提出基于高斯混合模型的不确定数据流处理，并对时间进行属性化处理。第 3 节详细描述 cumicro 算法的各个步骤。第 4 节提供实验结果及其分析。第 5 节对全文做总结并指出后续研究方向。

2 相关研究工作

目前，对不确定数据流的研究大多是基于离散型随机变量模型开展的，针对连续型随机变量模型的研究相对较少，主要是由于前者更利于计算机存储和运算[2]。Graham 等人利用抽样和直方图操作，针对连续型随机变量模型提出了多种基本算法[3]，但该模型面临的主要问题在于误差累积，导致最终结果可能不精确。使用连续型随机变量表示不确定数据，相对于离散型模型，更接近于真实世界。这类研究中，较为经典的是 CLARO 项目中使用的不确定数据流模型[4]，它使用高斯混合模型描述不确定数据流中的不确定数据（也称概率数据）。高斯混合模型是一种连续型概率分布模型，其概率密度分布理论上可以无限地近似任何其他分布，同时，其占用存储空间较小,可以方便地表示数据的不确定性。高斯混合模型具有较好的数学性质，其线性特性得到了证明,并被作为概要结构来存储数据流概要[5]。

在数据流聚类研究方面，国内杨宁等提出了一种基于时态密度的倾斜分布数据流聚类算法[6]，该算法只能处理欧氏空间单数据流，但在实际应用中,分布式环境下多数据流相互影响,相互作用,越来越多的数据流存在于非欧氏空间。陈华辉等利用数据流的遗忘特性来对数据流进行压缩,建立一个比整个数据流的数据规模小得多的概要数据结构来保存数据流的主要特征，提出了基于小波概要的并行数据流聚类[7]，但损失了数据的准确度。张晨等主要面向含存在级不确定性的不确定数据流的聚类问题，提出了一种不确定数据流聚类算法——EMicro 算法[8]。公茂果等提出了复杂分布数据的二阶段聚类算法[9]，该算法主要适用于复杂分布的静态数据聚类问题。朱林等针对静态的文本和基因

等高维数据，利用模糊可扩展聚类框架,与熵加权软子空间聚类算法相结合提出了一种基于数据流的软子空间聚类算法[10]。 以上的算法都有其各自应用的局限性，本文算法的应用与以上各算法有所区别,主要是考虑存储空间的影响，针对目前不确定数据流聚类在概要结构设计和时间的演化分析上并不完善，较难发现时间维度上的聚簇，进行改进。

国外较为典型的是 Aggarwal 等人提出的 umicro 算法[11]。该算法包含了两个关键技术，一是，模型使用了在线形成微簇和离线处理微簇两步聚类的方法，高效地处理数据流。微簇是聚类的中间产物，可以视为一个临时的聚簇。二是，模型对数据流中的时间属性进行了考虑，提出了时间演化数据流聚类的概念，提供了不同时间片段的聚类结果对比。后续其他的研究工作，如 SOStream 聚类算法[12]的提出，以及基于密度的数据流聚类方法 dDenStream[13]对数据进行挖掘，也都是在此基础上的改进。以上的算法，可以发现某一个时间段中的聚簇，但不能确定聚簇的时间范围；本文中算法将时间直接作为数据属性，可以直接查询某个时间维度的聚簇。

本文针对当分布特征较为复杂或当精度要求很高时，往往需要存储较多的数据点才能刻画出分布特征，且导致存储空间会成倍增长的问题；使用高斯混合模型描述了原始的不确定数据的不确定性，仅需要保存不同组件的描述信息即可，基于高斯混合模型的连续型随机变量的表示方法可以更好利用存储空间，完成对真实情况的逼近。

3 不确定数据流聚类算法框架

本文在经典的 umicro[11]不确定数据流聚类框架的基础上，设计了一种基于高斯混合模型表示的不确定数据流聚类算法 cumicro,算法框架如图 1 所示。该方法实现了增量式的在线聚类操作，形成一系列的微簇结构作为中间产物，通过对微簇进行离线加工，以获得最终的聚类结果。

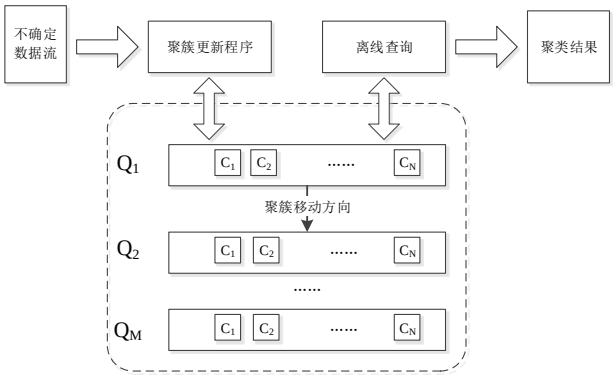

Fig. 1 A framework for clustering uncertain data streams

图 1 不确定数据流聚类框架

3.1 基于高斯混合模型的不确定数据流处理

与以往基于离散型随机变量表示的不确定数据的研究不同，本文使用高斯混合模型描述了原始的不确定数据的不确定性。高斯混合模型在表示不确定数据流上的优势体现在三个方面：(1)混合模型的每个组件都是高斯模型，高斯模型普遍存在于现实的分布中，可以较为贴切地表示真实数据，经过线性组合，高斯混合模型几乎可以近似任何其他形式的连续型分布。(2)高斯混合模型存储方便，针对模型中的每个组件，只需要存储几组相关参数即可。(3)高斯混合模型可以表示多维数据，即可以使用一个单一的多维高斯混合模型，对原有的多组属性进行统一存储。高斯混合模型的表示形式如下：

$$p(x|\pi,\mu,\sigma)=\sum_{i=1}^{r}\pi_i g(x|\mu_i,\sigma_i) \qquad (1)$$

$$g(x|\mu_i,\sigma_i)=\frac{1}{(2\pi)^{\frac{D}{2}}|\sigma_i|^{\frac{1}{2}}}\exp\left\{-\frac{1}{2}(x-u_i)'\sigma_i^{-1}(x-u_i)\right\} \qquad (2)$$

公式（1）中，高斯混合模型的分布函数 p 由

多个正态分布 $g(\boldsymbol{x}|\boldsymbol{\mu}_i,\boldsymbol{\sigma}_i)$ 叠加构成，其中每一个正态分布 $g(\boldsymbol{x}|\boldsymbol{\mu}_i,\boldsymbol{\sigma}_i)$ 称之为组件。模型中，$\boldsymbol{\pi}$ 为表示不同组件的权值的向量，$\boldsymbol{\mu}$ 为表示不同组件的平均值的向量，$\boldsymbol{\sigma}$ 为不同维度间的协方差矩阵。取值上，设原始数据为 D 维向量，组件共有 r 个，那么 $\boldsymbol{\pi}$ 为 $1*r$ 的向量，$\boldsymbol{\mu}$ 为 $D*r$ 的矩阵，$\boldsymbol{\sigma}$ 为范数为 r 的向量，其中每个元素为 $D*D$ 的相关系数矩阵。经过模型转换，本文所讨论的不确定数据流就变成了符合高斯混合模型的元素序列，每一个元素可以记为 $(t,\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\sigma})$，其中 t 为时间戳。

传感器后台数据库中存储的数据，是根据传感器节点的采样时间间隔获取不确定数据流中的一部分数据进行存储；而真实的数据流每时每刻源源不断的高速到达，其数据量远远大于传感器后台数据库中存储的数据。本文通过 EM 算法[14]，获得模型组件的个数，从而将其表示为高斯混合模型的形式，即转化为真实的数据流状态，再通过概要结构进行存储，最后对其进行相应的聚类。

3.2 多级队列概要结构

本文的概要结构如图 1 虚线框中所示，与文献[15]所论述的倒金字塔式的概要结构相似，采用多级队列的存储结构，以对高斯混合模型变换后的元素进行存储，上层队列的元素通过一定的规则合并后加入到下层，这样一来，对于近期的数据，可以获得尽量准确的聚类结果，但随时间的推移，生成的微簇结构中原始数据个数越来越多，元素的时间跨度越来越大，聚类的准确度会逐渐变低，在文中采用时间属性化的方法进行处理。

概要结构中共有 M 层队列，从顶层到底层依次为 $Q_1, Q_2, Q_3 \cdots\cdots Q_M$，每层可以存储 N 个微簇。在结构上，整体概要为 $(Q_1,Q_2,\cdots,Q_M)$，其中 Q_i 为概要的第 i 层，每层的结构为一个队列，每一层的队列结构定义为 $(C_1,C_2,\cdots,C_N)$，C_i 为初步聚类形成的微簇，微簇的结构包含一个高斯混合模型和时间信息，记为 (π,μ,σ,t,Tau,n)。其中 $\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\sigma}$ 是确定高斯混合模型的基本要素，$\boldsymbol{\pi}$ 为高斯混合模型的权值向量，$\boldsymbol{\mu}$ 为模型的均值向量，$\boldsymbol{\sigma}$ 为方差向量，t 和 Tau 是确定簇中元素的时间分布，t 是时间标签的均值，Tau 是时间标签的方差，n 为参与微簇形成的组件数目之和，微簇存储了后续聚类所需信息，底层存储了全局的聚簇状态。这里高斯混合模型不仅表示原始数据，还可以表示聚类中间形成的簇。

每一时刻，由多元素 $(t,\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\sigma})$ 构成的数据流首先到达概要结构的顶层，系统将判断该组元素是否属于顶层的某一个微簇，考虑是否进行合并、创建微簇或丢弃。如果某层的微簇已满，则合并其中的两个微簇，并将他们加入到下一层中。如果底层已满且有新聚簇到达时，则进行一定的舍弃操作，舍弃对当前影响最小的微簇。

3.3 时间的属性化处理

大多数据流聚类方法多使用时间戳的方式记录数据的到达时间，这种处理方式的好处是在聚类的过程中便于就近合并聚簇，但在查找相似聚簇时，得到的两个 GQFD 距离[16]最近聚簇可能有较大的时间跨度，这使得最终得到的聚簇的时间跨度难以控制；此外，这种方式只能给出聚类结果在不同时间片段上的投影，无法真正发现时间维度上的聚簇。可以考虑将时间作为一个属性参与聚类计算可以使问题得到解决，引入时间作为聚类的一个判定属性时，当时间间隔大时，聚簇的相似度较小；同时，由于每个聚簇都会包含时间信息，最终得到的聚类结果可以直观的获得聚簇在时间维度上的信息。将时间作为属性参与运算的方式称之为时间的属性化。

为了对时间进行属性化处理，矩阵 μ 均增加一列，σ 中元素个数不变，但每个相关系数矩阵会增加一行和一列。μ 增加的列为 t，σ 增加的元素为 Tau。为了便于计算，处理时，假设时间作为随机变量与其他的随机变量是独立的，故非主对角线增加的元素均为 0。对于用高斯混合模型处理后的每一个 σ_i 协方差矩阵，记扩展后为 σ_i'。

$$\mu_i' = \begin{pmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_n \\ t \end{pmatrix}, \sigma_i' = \begin{pmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1n} & 0 \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2n} & 0 \\ \vdots & \vdots & \ddots & \vdots & 0 \\ \sigma_{n1} & \sigma_{n2} & \cdots & \sigma_{nn} & 0 \\ 0 & 0 & \cdots & 0 & Tau \end{pmatrix} \quad (3)$$

初始值选取问题上，t 直接选用当前时间即可；Tau 控制了时间维度上聚类的快慢，其取值需要多方面考虑：如果取值过大，那么在时间维度上会形成较大的聚簇，取值过小会影响聚簇的形成，需要一个合适的 Tau 值来避免以上两种情况的发生。考虑到时间属性和其他属性的相互独立性，无法通过其他属性对时间属性进行推测，同时，实际应用中时间和其他数据计量单位往往不相同，Tau 的取值最好由用户指定，但这样却又带来有一定的随意性。因此，可以使用以下条件对 Tau 的取值进行约束。首先，由于协方差矩阵 σ_i' 为正定阵，所以 $Tau > 0$；其次，在若干数据到达之后，可以获得这些数据之间时间差的最大值和最小值，作为参数 Tau 的合理参考；本文时间属性中 Tau 的取值，需要用户根据聚类快慢的需求进行指定。

4 基于高斯混合模型的聚类

聚类过程，就是对概要结构中的微簇进行动态维护的过程。整个聚类过程包括概要结构更新、微簇合并和微簇简化等几大步骤。

4.1 概要结构更新

概要结构更新具体如算法 1 所示，其中 S_C 是一个存储微簇的临时结构，主要是用于向下层传递用于更新的数据，使用队列的形式存储。G 是指概要的当前状态，t 为当前时刻，相关度阈值 κ 控制每层聚类粒度变化的梯度，K 越小，距离的限制越严格，数据聚合速度越慢，不同层之间微簇的粒度变化越慢。参数 U 用于控制每个高斯混合模型中组件的个数，当组件个数超过 U 时，则需要对组件进行合并。

算法 1： 概要结构更新算法 **SketchUpdate**

输入： 当前状态 G，时刻 t，新到达的数据记为 X，相关度阈值 K，单个混合高斯模型组件个数上限为 U

输出： 下一个状态 G'

1. $i \leftarrow 1$，将 X 扩展为微簇，并加入微簇集合 S_C 中
2. **for** 状态 G 的每一层 i
3. **if** S_C 为空，返回 G 并结束
4. **if** Q_i 为底层
5. **for** S_C 中的每一个微簇 C
6. **if** 底层 Q_M 数据未满，则直接将 C 加入到 Q_M 中
7. 找到该层中与 C 的签名二次型距离 GQFD 距离最小的簇 C_t，求出 C 与 C_t 的距离 d
8. **if** $d < K$ 则调用 $CMerge(C_t, C, U)$，合并 C 和 C_t，Q_M 重新排序
9. 返回 G 并结束
10. **else**
11. **for** S_C 中的每一个微簇 C
12. **if** Q_i 未满，直接加入微簇 C
13. **else**
14. 找到该层中与微簇 C 的 GQFD 距离最小

的簇C_t，求出C与C_t的距离d

15. 找到本层间距最小的簇对C_m与C_n，以及他们之间的距离d'

16. **if** $\min(d,d')>K$

17. 将Q_i队列尾部元素放入微簇集合S_C中

18. C相对于本层为新点，将其放入队列Q_i首部

19. **else**

20. **if** $d<K$ 调用$CMerge(C_t,C,U)$合并C和C_t

21. **else** 调用$CMerge(C_m,C_n,U)$，将结果放入微簇集合S_C中

22. Q_i进行排序

算法 1 要进行微簇之间距离的估计，直接采用了签名二次型距离 GQFD[16]。这种方法有较高的效率，适合于流处理，同时具有对称性和满足三角不等式等良好性质，适合于相似度估计[11]。

更新后的概要结构具有以下的存储特性：首先，越接近底部聚簇包含的簇粒度越大；其次，对于Q_i和其下层Q_{i+1}，Q_i簇中时刻t的均值大于Q_{i+1}层簇的时刻t的均值。每次更新时，时间戳最小的元素如果不能合并，会被直接被放于队首，如果可以合并，则会在完成之后重新排序，故队列之间和队列之内的微簇的时间均值t总保持有序。

4.2 微簇合并

微簇的合并是算法 1 的子过程。由于高斯混合模型本身具有很好的线性特性[3]，模型之间的合并实际上是参数向量$\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\sigma}$的扩展。假设微簇$C_1$，$C_2$合并之后形成$C_3$，记微簇$C_1$，$C_2$的概率密度函数分别为$f_1(x)$，$f_2(x)$，$m$，$n$分别为$C_1$，$C_2$中原始数据的个数，由文献[3]可知，聚簇$C_3$的概率密度满足式(4)：

$$f_3(x)=\frac{n}{m+n}f_1(x)+\frac{m}{m+n}f_2(x) \qquad (4)$$

由于$f_1(x)$，$f_2(x)$的每个组件均为加权的正态分布，线性叠加形成的$f_3(x)$也是一个高斯混合模型，其组件为簇C_1，C_2的组件之和。合并操作不改变新簇每个组件中的μ和σ，只改变新簇的组件的权重，新簇权重向量π根据来源不同分为两部分，分别乘以系数$n/n+m$和$m/n+m$。

微簇合并中存在的一个问题是组件的“碎片化”：经过多次合并，组件数目并没有减少，但每个组件的权重不断减小。由于 GQFD 方法以微簇组件为基本计算单位，过多的碎片会大大降低计算效率。这里，设定微簇的组件数目上限为U，当组件数目超过U时，合并均值μ最为接近的两个组件，形成的新组件权重π为原有两个组件权重之和，均值μ为加权均值，方差为原始的最大方差。这种处理方法虽然损失了方差上的精度，但较好保存了数值特征。具体算法如下：

算法 2：微簇合并算法$CMerge(C_P,C_q,U)$

输入：微簇$C_p(\pi_{p,}\mu_{p,}\sigma_{p,}t_{p,}T\ a\ \mu n_p)$和微簇$C_q(\pi_{q,}\mu_{q,}\sigma_{q,}t_{q,}T\ a\ \mu n_q)$，层$Q_i$，组件上限$U$

输出：合并生成的新簇C'

1. $p\leftarrow n_p/(n_p+n_q)$，$q\leftarrow n_q/(n_p+n_q)$，$k_p$和$k_q$为$C_p$ C_q中组件个数

2. **for** $i\in[1,k_p]$

3. $C'.\pi_i\leftarrow p\pi_i$，$C'.\mu_i\leftarrow\mu_i$，$C'.\sigma_i\leftarrow\sigma_p$

4. **for** $j\in[1,k_q]$

5. $C'.\pi_{k_p+j} \leftarrow p\pi_j, C'.\mu_{k_p+j} \leftarrow \mu_j$， $C'.\sigma_{k_p+j} \leftarrow \sigma_j$

6. $C'.n \leftarrow n_p + n_q$

7. **while** 微簇 C 中的组件数大于 U

8. 找到 C 中最近的两个组件 $g_1(\pi_1,\mu_1,\sigma_1)$ 与 $g_2(\pi_2,\mu_2,\sigma_2)$

9. $g'(\pi',\mu',\sigma')$ 为新组件

$$\pi' \leftarrow \pi_1 + \pi_2,\ \mu' \leftarrow \frac{\pi_1\mu_1 + \pi_2\mu_2}{\pi_1 + \pi_2},$$

$$\sigma' \leftarrow \max(\sigma_1,\sigma_2)$$

10. 使用 g' 代替 g_1，g_2

算法 2 中，步骤 1 到 6 完成了两个微簇的组件的合并，7 到 10 完成了组件的简化。

4.3 聚类结果查询

聚类的最终结果可以通过对已有的概要结构中的聚簇进行离线聚类获得。由于每层队列中微簇之间所属的粒度不同，每个微簇实际上就是一个聚簇结果，且该聚簇结果是包含时间属性的。由前所知，概要结构在层次上是有时间顺序的，所以通过常数次的扫描，可以获得任意一个时间段下的聚类结果。

5 实验分析

5.1 实验设置

由于 umicro 算法[11]在处理数据流聚类方面的高效性，本节以该算法为基准算法来验证 cumicro 算法的性能，而 SOStream 算法和 dDenStream 算法主要是基于密度的聚类,与本文的聚类有本质不同，故不与比较。在不确定数据源方面，由于目前已有的一些开源数据集的数据分布不能事先预知，从某种程度上会给后面的实验结果的验证带来了困难；因此，使用了传感器在猪舍中监测到的真实环境数据集进行了仿真实验。在三个育肥猪舍中，布置了温度、湿度、氨气、硫化氢传感器，采样频率为 10 分钟一次，取一个月的监测数据，并将时间作为多维空间中的一个属性作为实验数据。由于饲养密度、生猪品种等的不同，每个猪舍的传感器采集到的数据总体上服从各自的函数分布，数据从总体上来讲可分为三大类，但不可避免的在某些情况下，会出现相同或是相近的数据，这些数据正好可以对算法聚类的有效性进行检验。

5.2 聚类评价方法的选取

常用的聚类有效性评价方法有外部评价法、内部评价法和相对评价法[17]。外部和内部评价法均基于统计测试，F-measure 是一种外部评价法，它组合了信息检索中查准率(precision)与查全率(recall)的思想来进行聚类评价。

$$F(i) = \frac{2*P*R}{P+R} \qquad (5)$$

其中，P 代表准确率，R 代表召回率；对分类 i 而言，哪个聚类的 F-measure 值高，就认为该聚类代表分类 i 的映射。换句话说，F-measure 可看成分类 i 的评判分值[18]。

5.3 参数 K 的影响

阈值 κ 的选取与数据自身属性有较大的相关性。直观上，该阈值应当能尽量区分原本属于不同分类的元素，同时不影响同类元素的合并。使用 F-measure 作为聚类效果的评估方式，以此分析不同 K 值下的聚类结果，ratio of F-measure 值越大，聚类效果越好。

κ 决定了聚类的效果，两个微簇间的相似度越高，意味着聚成一类的可能性越大。实验的数据集，在 Tau 取值为 0.025 时,采用 EM 算法进行初始加工，将原始数据加工为高斯混合模型表示的不确定数

据，聚类完成后通过外部标签对正确率进行评价。实验针对 K 值在 0.5 到 2.5 区间内每隔 0.5 进行一次计算，最终得到如图 2 所示结果。K 值在 1.82 附近时聚类取得最好效果，但 K 大于 1.82 时聚类效果显著下降，这是由于阈值过大导致合并条件过于苛刻，以至于同类元素被误判为属于不同类别。此外，K 大于 1.82 后的 F-measure 值明显下降，当 K 大于 2 之后，始终为 0.17，上述两种情况下的聚类效果较差，主要是由 F-measure 评价机制引起的。

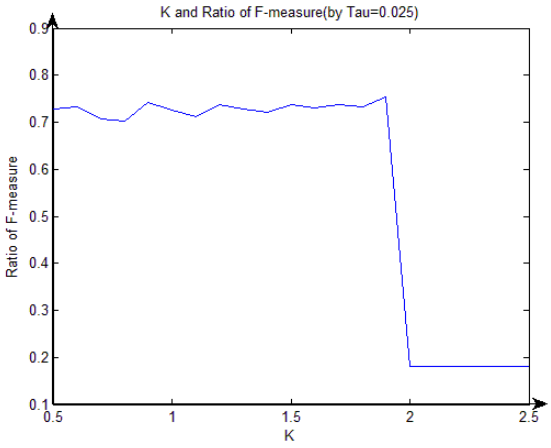

Fig.2 Different F-measure result by changing k

图 2 数据集在不同 K 值下 F-measure 的表现

5.4 参数 *Tau* 的影响

实验数据集在 K 取值为 1.817 时，采用 EM 算法进行初始加工，将原始数据加工为高斯混合模型表示的不确定数据，聚类完成后通过外部标签对正确率进行评价。如图 3 所示，当 *Tau* 取值在小于 0.27 时，F-measure 值取得较好的结果，当 *Tau* 取值在大于 0.27 时，F-measure 值急剧下降；当 *Tau* 取值大于 0.325 时，相关性减弱，F-measure 值不再发生变化，聚类的效果较差。

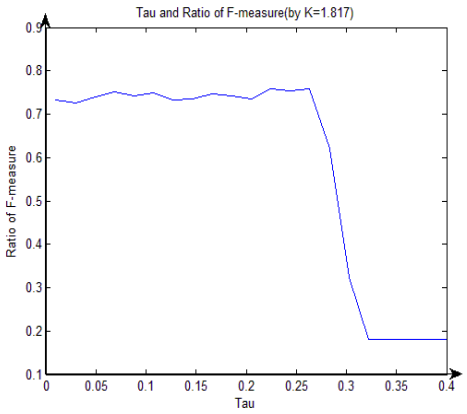

Fig.3 Different F-measure result by changing *Tau*

图 3 数据集在不同 *Tau* 值下 F-measure 的表现

5.5 边界系数 D 的影响

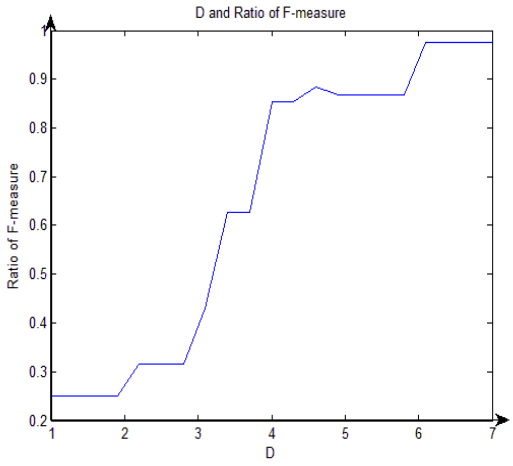

Fig. 4 Different F-measure result by changing D

图 4 数据集在不同 D 值下 F-measure 的表现

采用 umicro 的数据流聚类方法，获得如图 4 所示的试验结果，从图 4 中可以看出，随着 D 取值的增大， F-measure 值也在增大;这是因为 D 越大，范围越宽，对应的划为同一类的可能性越大，越容易聚类，当试验数据的 D 取值为 6 以上时，F-measure 值可达到 0.96 以上。

5.6 UMicro 算法与本文 cumicro 算法的比较

上述实验可以看出，本文提出的 cumicro 算法的 F-measure 值受参数 K 和 *Tau* 综合影响，K 和 *Tau* 越

小，越容易聚类，因为 K 和 Tau 反应了相关性；umicro 的 F-measure 值受 D 的影响，因为 D 控制了聚类发生的边界，边界值 D 越大，越容易聚类。为了保持实验的合理性，将 K 和 Tau 值尽量取小，将 D 值尽量取大，以同时保持两种算法的优越性，但又不可太过，以免影响到算法本身的合理性。综合比较图二、图三、图四的关系曲线，考虑到 D 一般取到 6 左右会比较好地反应数据聚类的特点，因此，这里 D 取 6；考虑到 K 在大于 1.8 时，cumicro 基本无效，Tau 取值在小于 0.27 时，F-measure 值取得较好的结果，故在下面的比较实验中 K 取 1.5，Tau 取 0.005。

图 5 的横坐标为 σ，从图 5 可以看出，随着 σ 的增大，cumicro 算法的趋势是整体下降的，这是因为数据在空间上相距较近，重叠的数据增多；若 σ 非常小，cumicro 算法有误判的可能，准确度不如 umicro 算法；当 σ 非常大，umicro 无法区分不同的分类，此时，本文提出的 cumicro 算法的优势体现出来了，可以正常工作。

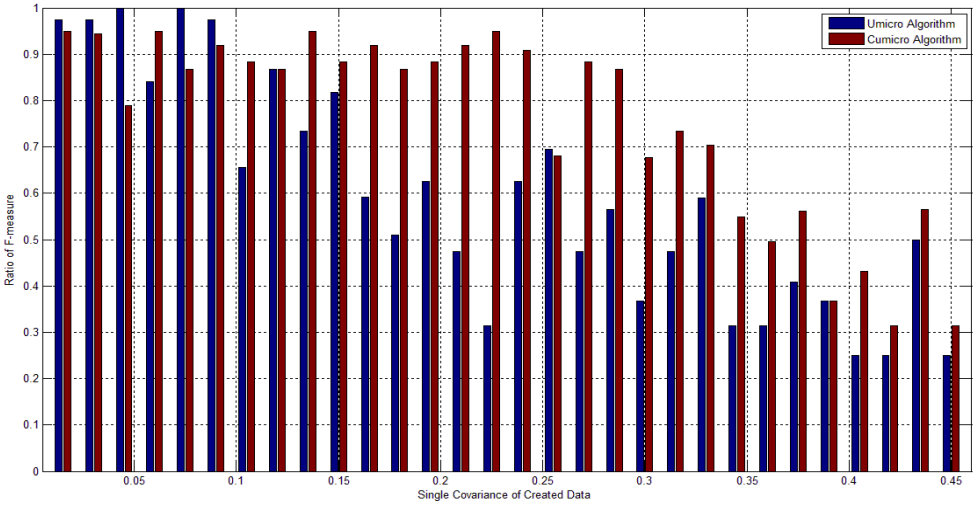

Fig. 5 Comparison between umicro algorithm and cumicro algorithm

图 5 两种算法的聚类准确率比较

表 1 给出了本文算法 cmicro 与 umicro 算法的对比。

Table 1 Comparison Between Umicro Algorithm And Cumicro Algorithm

表 1 umirco 算法和本文算法 cmicro 的对比

概率聚类方法 / 对比项	Umicro 算法	基于混合高斯模型的概率流聚类方法
算法框架	类似于 Clustream 的在线聚类和离线聚类两步	相同
概要结构	类似于 Clustream 的多层结构	类似，多层队列
输入数据表示	使用基于元组级不确定性的不确定数据表示	使用高斯混合模型表示
最小存储结构	使用中心位置和半径表示聚簇的微簇结构	使用高斯混合模型表示的微簇结构
对时间的处理	离线聚类中统一处理，可以发现某一个时间段中的聚簇，但不能确定聚簇的时间范围	将时间直接作为数据属性，可以直接查询某个时间维度的聚簇

6 结论

本文在 umicro 算法的基础上，使用高斯混合

模型作为不确定数据的表示形式，通过对时间进行属性化的处理，设计了新的概要结构的动态维护方法，估计了不同 GMM 簇的相关性，从而实现了针对连续型随机变量的不确定数据流聚类算法，并通过实验验证了该方法的有效性。这种聚类方法有助于挖掘不确定数据流上的时间特性，同时由于采用基于密度的聚类方法，避免了传统基于划分的聚类中较难发现非球状聚簇的问题。这种方法仍然存在舍弃和合并规则有可能带来精度损失等问题，这也是之后的研究目标。

参 考 文 献

[1] Akbarinia, Reza ; Masseglia, Florent.Fast and exact mining of probabilistic data streams. In: Machine Learning and Knowledge Discovery in Databases - European Conference.Prague: ECML PKDD,2013.493-508.

[2] Aggarwal, C.C. and P.S. Yu, A survey of uncertain data algorithms and applications. Knowledge and Data Engineering, IEEE Transactions on, 2009. 21(5): 609-623.

[3] Cormode, G. and M. Garofalakis, Sketching probabilistic data streams, in Proceedings of the 2007 ACM SIGMOD international conference on Management of data. Beijing, China:ACM, 2007：281-292.

[4] Tran, T.T., et al., CLARO: modeling and processing uncertain data streams. The VLDB Journal—The International Journal on Very Large Data Bases, 2012, 21(5): 651-676.

[5] Song, M. and H. Wang. Highly efficient incremental estimation of gaussian mixture models for online data stream clustering[J]. Intelligent Computing: Theory and Applications, 2005, 5803(1): 174-183.

[6] 杨宁，唐常杰，王悦，陈瑜，郑皎凌.一种基于时态密度的倾斜分布数据流聚类算法.软件学报, 2010,21(5):1031−1041.

[7] 陈华辉，施伯乐，钱江波.陈叶芳.基于小波概要的并行数据流聚类.软件学报, 2010,21(4):644−658.

[8] 张晨，金澈清，周傲英.一种不确定数据流聚类算法. 软件学报, 2010,21(9):2173−2182.

[9] 公茂果，王爽,马萌，曹宇等.复杂分布数据的二阶段聚类算法.软件学报, 2011,22(11):2760−2772.

[10] 朱林, 雷景生，毕忠勤，杨杰.一种基于数据流的软子空间聚类算法.软件学报,2013,24(11):2610−2627.

[11] Aggarwal CC. On high dimension projected clustering of uncertain data streams. In: Proc. of the 25th Int'l Conf. on Data Engineering. Shanghai: IEEE, 2009. 1152−1154.

[12] Charlie Isaksson, Margaret H. Dunham, Michael Hahsler. SOStream: Self organizing density-based clustering over data stream . In: Proc. of the 8th International Conference. On Machine Learning and Data Mining in Pattern Recognition. Berlin: IEEE, 2012. 264-278.

[13] Kumar, Manoj, Sharma, Ashish.Mining of data stream using "dDenStream" clustering algorithm. In: Proc. of the 2013 Int'l Conf in MOOC. on Data Engineering. Jaipur: IEEE, 2013.315-320.

[14] Arnaout, A. Automatic threshold tracking of sensor data using expectation maximization algorithm. In: Proceedings of the 2011 11th International Conference on Hybrid Intelligent Systems. Malacca : IEEE, 2011 .551-554.

[15] Aggarwal CC, Yu PS. A framework for clustering uncertain data streams. In: Proc. of the 24th Int'l Conf. on Data Engineering. Cancún: IEEE, 2008. 150−159.

[16] Beecks, C., et al. Modeling image similarity by gaussian mixture models and the signature quadratic form distance. in Computer Vision (ICCV), 2011 IEEE International Conference on. Barcelona: IEEE, 2011.1754 - 1761.

[17] Chen, Weijin,Dong, Huailin,Wu, Qingfeng. Research on fuzzy clustering validity. In: 2010 WASE Global Congress on Science Engineering. Yantai: GCSE,2011.174-182.

[18] THEODORIDIS S,KOUTROUBAS K. Pattern recognition. Beijing: Academic Press,2008.

曹振丽，手机：1850，1939，749

E-mail: caozhenli2004@163.com