ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.21, No.10, October 2010, pp.2513−2523
doi: 10.3724/SP.J.1001.2010.03763

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

自适应的软子空间聚类算法*

陈黎飞[1+], 郭躬德[1], 姜青山[2]

[1](福建师范大学 数学与计算机科学学院,福建 福州 350108)

[2](厦门大学 软件学院,福建 厦门 361005)

Adaptive Algorithm for Soft Subspace Clustering

CHEN Li-Fei[1+], GUO Gong-De[1], JIANG Qing-Shan[2]

[1](School of Mathematics and Computer Science, Fujian Normal University, Fuzhou 350108, China)

[2](Software School, Xiamen University, Xiamen 361005, China)

+ Corresponding author: E-mail: clfei@fjnu.edu.cn, http://math.fjnu.edu.cn/newmath/Article/ShowArticle.asp?ArticleID=741

Chen LF, Guo GD, Jiang QS. Adaptive algorithm for soft subspace clustering. *Journal of Software*, 2010, 21(10):2513−2523. http://www.jos.org.cn/1000-9825/3763.htm

Abstract: Soft subspace clustering is a key for high-dimensional data analysis. The existing algorithms usually require users to estimate some key global parameters in advance, and ignore the optimization of subspaces. A novel objective function, to be optimized by the soft subspace clustering algorithms, is proposed in this paper by taking into account both minimization of the compact subspace clusters and maximization of the subspaces in which the clusters exist. Based on this, a new locally feature weighting scheme is derived, and an adaptive algorithm for *k*-means type soft subspace clustering is presented. In the new algorithm, the optimal values of parameter are automatically computed, according with the dataset and its partitions. Experimental results carried out on some real-world and synthesis datasets demonstrate that the proposed method significantly improves the accuracy as well as the stability of the clustering results.

Key words: clustering; high dimensional data; subspace; feature weighting; adaptability

摘 要: 软子空间聚类是高维数据分析的一种重要手段.现有算法通常需要用户事先设置一些全局的关键参数,且没有考虑子空间的优化.提出了一个新的软子空间聚类优化目标函数,在最小化子空间簇类的簇内紧凑度的同时,最大化每个簇类所在的投影子空间.通过推导得到一种新的局部特征加权方式,以此为基础提出一种自适应的*k*-means 型软子空间聚类算法.该算法在聚类过程中根据数据集及其划分的信息,动态地计算最优的算法参数.在实际应用和合成数据集上的实验结果表明,该算法大幅度提高了聚类精度和聚类结果的稳定性.

关键词: 聚类;高维数据;子空间;特征加权;自适应性

* Supported by the National Natural Science Foundation of China under Grant No.10771176 (国家自然科学基金); the Fujian Provincial Natural Science Foundation of China under Grant No.2009J01273 (福建省自然科学基金); the Scientific Research Foundation for the Returned Overseas Chinese Scholars, Ministry of Education of China under Grant No.[2008]890 (国家教育部留学回国人员科研启动基金); the Key Scientific Research Project of the Higher Education Institutions of Fujian Province of China under Grant No.JK2009006 (福建省省属高校科研专项重点项目)

Received 2009-04-24; Revised 2009-08-12; Accepted 2009-10-19

中图法分类号: TP311 文献标识码: A

聚类是数据挖掘研究的一个重要手段,研究者已提出多种聚类算法[1].然而在许多实际应用中,数据具有很高的维度(也称为特征或属性).例如,文本挖掘中由 VSM(向量空间模型)[2]表示的文档向量可能具有几百甚至上千个特征.受“维度效应(the curse of dimensionality)”[3,4]的影响,当用传统算法聚类如此高维的数据时,有效性大为降低.高维数据的处理问题是目前数据挖掘的挑战性任务之一[3−5].

高维数据空间中往往存在许多不相关的属性,使得要寻找的目标类只存在于某些低维子空间中,而不同的簇类其关联的子空间通常也是不一样的[4,6].在高维空间挖掘隐藏在不同低维子空间中簇类的过程,称为子空间聚类(subspace clustering).现有多种类型的子空间聚类算法[4],其中,投影聚类(projective clustering)由于对数据维度数目增长不敏感等优点,自 Aggarwal 等人[6]提出开始就获得了广泛的关注.其核心思想是,给定簇数目 K,在一个类 k-means(或 k-medoids)算法过程中对数据集进行划分的同时,搜索每个划分所在的最佳投影子空间.最佳子空间通过局部特征加权(locally feature weighting)技术[7]实现,对于每个划分,各维度被赋予不同的权重,权重越大表示特征越重要,或与该划分的关联性越强.

根据加权方式的差异,现有算法可分为硬子空间(hard subspace)和软子空间(soft subspace)聚类两种方法[8],后者给维度赋予[0,1]区间的权值,表示维度与对应划分之间“模糊的”关联度,是近年来较为活跃的一个研究方向.现有算法在定义加权方式时都引入了一些难以确定的参数,例如 FWKM[9]的β和δ,LAC[7]的 h,FSC[10]的α和ε以及 EWKM[8]的γ等,用于计算每个划分的投影子空间.实际应用中,以上算法的参数都需要用户给定.另外,这些参数都被定义为全局的(不同的簇类使用统一的加权参数),因而无法根据数据集及其各异的簇结构特点进行自动调节,这些都导致了算法对不同聚类问题泛化能力的降低.此外,现有算法在聚类过程中大多着重于数据集划分的优化,而忽略了投影聚类问题的另一方面:簇类所在子空间的优化.这是不完备的.

本文提出一种自适应的算法 ASC(adaptive soft-subspace clustering),用于对高维数据进行软子空间聚类.ASC 基于经典的 k-means 聚类过程,实现数据集划分和投影子空间两方面的优化;除簇数目 K 以外,ASC 不需要用户设置额外的参数,而是在算法的每个迭代步骤根据数据集及其划分的特点,动态地计算优化的加权参数,为每个簇类确定其最佳的投影子空间.在合成数据和实际应用数据集上的实验结果表明,ASC 算法有效提高了聚类精度和聚类结果的稳定性.

本文第 1 节主要介绍背景知识和相关的研究工作.第 2 节给出具体的 ASC 算法描述.第 3 节介绍实验环境和实验结果分析.最后在第 4 节对本文进行总结并指出进一步的研究方向.

1 背景知识与相关工作

给定数据集 $DB=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_i,\ldots,\boldsymbol{x}_N\}$,其中,$\boldsymbol{x}_i=\langle x_{i1},x_{i2},\ldots,x_{ij},\ldots,x_{iD}\rangle$,称 $\boldsymbol{x}_i$ 为 $D(D>1)$维数据空间的第 i 个数据点($i=1,2,\ldots,N$).这里,$N(N>1)$表示数据点数目,并假设数据已作规范化处理使得所有的 $x_{ij}\in[0,1],j=1,2,\ldots,D$.一个硬聚类算法[1]将 DB 划分为 K 个子集的集合 $C=\{C_1,C_2,\ldots,C_K\}$,且$\forall k\neq l,1\leqslant k,l\leqslant K,C_k\cap C_l=\varnothing$,$C_k$ 称为 DB 的第 k 个簇($k=1,2,\ldots,K$),$K(K>1)$是给定的簇数目.

在投影聚类中,为识别某个簇类所处的子空间,可以采用维度加权的方法[4,7,11].设权重 w_{kj} 表示第 j 个维度与 C_k 的相关程度,其值越大,表示该维度对于 C_k 越重要.在硬子空间聚类算法中,如 PROCLUS[6]和 FINDIT[12]等,$w_{kj}\in\{0,1\}$,亦即只体现出维度是否与某个簇相关.这类算法通常需要事先给定相关维度的数目,例如 PROCLUS[6]的算法参数 l,这在实际应用中是很困难的.软子空间聚类算法,如 LAC[7],EWKM[8],FWKM[9]和 FSC[10]等,使用软特征加权方法使得 $w_{kj}\in[0,1]$且满足

$$\sum_{j=1}^{D} w_{kj}=1,\ k=1,2,\ldots,K \tag{1}$$

可以认为是硬子空间聚类算法的泛化.上述软子空间聚类算法的主要差异在于维度加权方法的不同.例如,FWKM 使用的权重计算公式[9]如下:

$$w_{kj}^{(\mathrm{FWKM})}=\left(\sum_{l=1}^{D}\left(\frac{\sum_{\boldsymbol{x}_i\in C_k}(x_{ij}-v_{kj})^2}{\sum_{\boldsymbol{x}_i\in C_k}(x_{il}-v_{kl})^2}\right)^{\frac{1}{\beta-1}}\right)^{-1} \tag{2}$$

这里,$\boldsymbol{v}_i=\langle v_{i1},v_{i2},\ldots,v_{iD}\rangle$表示 C_k 的簇中心,β为用户定义的加权参数.LAC[7]和 EWKM[8]等使用指数加权的方法,虽然避免了公式(2)中分母可能出现为 0 的情形,但同样也引入了难以确定的、却对聚类结果有重要影响的加权参数,如 LAC[7]中的 h 和 EWKM[8]的γ等.

这些加权方法体现了一个共同特点:维度权值的大小与数据点投影到该维度上的分布离散程度成“反比”[8,13],即数据集某个特征的取值越集中,那么其重要性就越高,这是现有投影聚类算法基于的共同假设[13].为了搜索每个簇类所处的最佳子空间,软子空间聚类算法[7−10,14]通常采用 k-means 型的算法结构,在经典的 k-means[1]基础上增加一个迭代步骤,为每个簇类计算与其相关联的维度权重集合.

给定一个数据集,为了获得一个好的聚类结果,需要对聚类质量进行评估,这是聚类有效性问题[15].在投影聚类中,各簇类可能存在于不同的投影子空间,这使得衡量投影簇类间的差异变得困难.一种可行的方法是转而评估每个投影子空间的“质量”,并在聚类过程中进行优化.EWKM[8]引入了维度“权重熵”因子,其目标是令投影子空间各维度权重分布具有最大的熵,这是基于维度权重服从某种概率分布的假设基础上.另外,该算法还存在如何在权重熵因子和扩展的 k-means 优化目标之间取得平衡的问题,正如实验结果所示(第 3.2 节),这些都降低了算法对不同聚类问题的泛化能力.DOC[16]从另一种角度评价投影聚类质量,认为一个好的投影簇类集合应使得每个簇类包含尽可能多的数据点,同时具有尽可能大的投影子空间.除引入难以确定的算法参数外,DOC 实际上简化了投影聚类问题,它根据数据点数目而不是簇内数据点的分布(已被广泛接受的一种度量指标便是“簇内紧凑度”[15])来衡量簇类的质量;其次,DOC 使用投影子空间的维度数目表示空间的大小,仅适用于硬子空间聚类.

本文提出一种衡量软子空间大小的新指标,与投影簇类“簇内紧凑度”度量相结合来定义新的目标优化函数.这两项指标构成了衡量投影簇类质量的两个相互制约的因素,高质量的投影聚类结果对应于二者间的一种平衡状态.新算法 ASC 以此为目标,在类 k-means 的聚类过程中最小化“簇内紧凑度”的同时,最大化每个簇类所在的投影子空间,以获得高质量的聚类结果.

2 ASC 算法

本节首先提出投影子空间等形式定义,在此基础上给出算法描述,并对算法的主要步骤和性能进行分析.

2.1 投影子空间

对于一个 D 维欧氏空间,D 个线性无关的 D 维向量可以构成空间的一组基.给定一组基,空间中任意一个数据点的坐标是确定的.因此,可以使用这样一组向量来代表一个空间.下面用矩阵 S_k 表示 C_k 所在的子空间:

$$S_k=\begin{bmatrix}\sqrt{w_{k1}} & & & \\ & \sqrt{w_{k2}} & & \\ & & \ddots & \\ & & & \sqrt{w_{kD}}\end{bmatrix}.$$

S_k 的 D 个行向量是线性无关的,组成了 $\mathcal{R}^D$ 的一组基.由于 S_k 是对角矩阵,因而只能表示坐标轴对齐型子空间(axis-aligned subspace)[17].本文不考虑非坐标对齐型(或称任意方向,arbitrary oriented[18])子空间,这是因为构成该型空间的特征通常是原始数据空间特征的某种线性组合,通过组合得到的新特征可解释性变差[4],从而限制了在一些领域(如文档聚类)的实际应用.

定义 1(子空间簇类). 子空间簇类 SC_k 是一个二元组 $SC_k=(S_k,C_k),k=1,2,\ldots,K$.

根据定义 1,下面也称 S_k 为 $\mathcal{R}^D$ 的第 k 个子空间.为检验 C_k 中的数据点是否在 S_k 表示的子空间中形成簇类,需要将数据点投影到该空间.

定义 2(投影). 设 $\boldsymbol{y}_i$ 表示 $\boldsymbol{x}_i$ 在 S_k 上的投影:

$$\boldsymbol{y}_i = \pi_{S_k}(\boldsymbol{x}_i) = \boldsymbol{x}_i \cdot S_k \tag{3}$$

定义 2 可以用以下推导来验证.考虑 S_k 中两个投影点 $\boldsymbol{y}_1$ 和 $\boldsymbol{y}_2$ 间的欧氏距离,有

$$\begin{aligned} dist_{S_k}(\boldsymbol{y}_1,\boldsymbol{y}_2) &= \sqrt{(\pi_{S_k}(\boldsymbol{x}_1)-\pi_{S_k}(\boldsymbol{x}_2))(\pi_{S_k}(\boldsymbol{x}_1)-\pi_{S_k}(\boldsymbol{x}_2))^T} \\ &= \sqrt{(\boldsymbol{x}_1-\boldsymbol{x}_2)S_kS_k^T(\boldsymbol{x}_1-\boldsymbol{x}_2)^T} \\ &= \sqrt{\sum_{j=1}^{D} w_{kj}(x_{1j}-x_{2j})^2}. \end{aligned}$$

上式最后一行恰恰是常用的加权欧氏距离度量[7−10,14].为了进行子空间之间的比较,我们基于矩阵 S_k 的迹(trace)来衡量第 k 个子空间的大小.S_k 的迹为

$$Trace(S_k) = \sum_{j=1}^{D}\sqrt{w_{kj}}.$$

性质 1. $Trace(S_k) \in [1,\sqrt{D}]$.

证明:求带约束条件 $w_{k1}+w_{k2}+\ldots+w_{kD}=1$ 的 $Trace(S_k)$极值问题,得到性质 1 结论. □

根据性质 1,可以定义标准化的子空间大小如下(将 $Trace(S_k)$变换到[0,1]区间).

定义 3(子空间大小). 记第 k 个子空间簇类的子空间大小为 $size(SC_k)$,

$$size(SC_k) = \frac{Trace(S_k)-1}{\sqrt{D}-1}.$$

2.2 目标优化函数

下面基于经典的 k-means[1]推导 ASC 算法.k-means 是一种 EM 型算法[19],它在迭代过程中不断更新数据集的划分,用以优化以下目标函数:

$$R_0(C,V) = \sum_{k=1}^{K}\sum_{\tilde{\boldsymbol{x}}_i \in C_k} \| \tilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{v}}_k \|^2.$$

k-means 在全空间搜索数据集的最优划分,记号 $\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{v}}_k$ 分别表示全空间中的第 i 个数据点和第 k 个划分的中心,$\|\cdot\|$表示 L_2 范数.为使 k-means 能够挖掘隐藏在不同子空间中的簇类,需要将 R_0 推广至子空间聚类.为此,将 R_0 中的 $\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{v}}_k$ 看作是原始数据点 $\boldsymbol{x}_i$ 和簇中心 $\boldsymbol{v}_k$ 投影在 S_k 中的结果,根据定义 2 和公式(3),目标函数可变换为

$$R_1(C,V,W) = \sum_{k=1}^{K}\sum_{j=1}^{D} w_{kj}\sum_{x_i \in C_k}(x_{ij}-v_{kj})^2.$$

R_1 就是常用的“误差测度(error measure)”[7].一个最小化 R_1 的算法可以获得 K 个具有最小“簇内紧凑度”的簇.ASC 算法的优化目标还包括最大化 K 个簇类所在的投影子空间,为此,在 R_1 的基础上引入定义 3 给出的子空间尺度衡量指标,构造新的优化目标函数如下:

$$J(C,V,W) = \sum_{k=1}^{K}\left(\sum_{j=1}^{D} w_{kj}\sum_{x_i \in C_k}(x_{ij}-v_{kj})^2 - h_k \times size(SC_k)\right).$$

其中,$h_k(h_k>0)$为平衡系数.上式中,由$-h_k$ 连接起来的前后两项相互制约,这是聚类有效性方法所需要的[15−17].例如,当 $size(SC_k)$取最大值时,$w_{k1}=w_{k2}=\ldots=w_{kD}=1/D$,那么 J 退化为 k-means 聚类目标函数.若考虑投影聚类则可以认为,此时簇的紧凑性达到最小;相反地,若 $size(SC_k)$取最小值,即除了一个数据维度的权重值为 1 外,其他维度的权重值为 0,这意味着数据点被投影到一个一维子空间中,因而该簇将具有最大的紧凑性.最好的聚类结果对应于这两个矛盾因素之间的一种平衡.在数值上,平衡系数 h_k 的作用是将前后两项变换到相同的取值范围,由于 $size(SC_k)$已作标准化处理(定义 3),故可设定

$$h_k = \begin{cases} \dfrac{1}{D}\sum_{j=1}^{D} X_{kj}, & \text{if } \sum_{j=1}^{D} X_{kj} > 0 \\ 1(\text{any constant}), & \text{otherewise} \end{cases}.$$

这里,引入了新记号 X_{kj}:

$$X_{kj}=\sum_{\boldsymbol{x}_i\in C_k}(x_{ij}-v_{kj})^2.$$

X_{kj}反映了C_k投影到第j个维度上数据点分布的离散程度.

2.3 算法过程

新算法ASC以J为优化目标,采用EM算法结构[19]来最小化J,算法描述如下:

Algorithm 1. ASC.

Input: the dataset $\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_N\}$ and the number of clusters K;

Output: $C=\{C_1,C_2,...,C_K\}$ and W.

Randomly choose K cluster centroids and set all initial weights to $1/D$;

Let p be the number of iteration, p=0; Denote V, W as $V^{(0)}$ and $W^{(0)}$, respectively.

Repeat

1. For each center $\boldsymbol{v}_k$, and for each point $\boldsymbol{x}_i$:

   Set $C_k=\{\boldsymbol{x}_i\mid k=\arg\min_{l=1,2,...,K} dist_{S_l}(\boldsymbol{v}_l,\boldsymbol{x}_i)\}$, where $dist_{S_l}(\boldsymbol{v}_l,\boldsymbol{x}_i)=\sqrt{\sum_{j=1}^{D}w_{lj}(x_{ij}-v_{lj})^2}$.

2. Compute the adaptive parameters using Eq.(6).

3. p=p+1;

4. Using Eq.(4) to calculate $V^{(p)}$;

5. Using Eq.(5) to calcalute $W^{(p)}$;

Until $\|V^{(p)}-V^{(p-1)}\|_\infty<\varepsilon$ and $\|W^{(p)}-W^{(p-1)}\|_\infty<\varepsilon$.

其中:$V=\{v_{kj}\}_{K\times D}$和$W=\{w_{kj}\}_{K\times D}$是两个矩阵;ε是一个很小的正实数,用于控制算法的终止,本文设定$\varepsilon=10^{-6}$.与现有算法[6−13,16]相比,ASC的特点首先体现在其自适应性上,ASC不需要用户输入加权参数,而是在第2步为由算法第1步生成的各数据集划分确定其不同的优化参数(见第2.4节);其次,ASC的目标还包括子空间的优化,这是现有其他算法所不具备的.

分析ASC算法过程,其迭代循环中的第1步对应于EM[19]的‘E’步骤,给定$V^{(p)}$和$W^{(p)}$将数据集划分为K个簇得到$C^{(p)}$;第4步和第5步对应于‘M’步骤,在给定$C^{(p)}$情况下计算$V^{(p+1)}$和$W^{(p+1)}$,而后在下一轮迭代的‘E’步骤重新划分数据集.根据EM的原理[19],每一步迭代都增大似然.在ASC中,这个目标对应于降低J的值.为了计算‘M’步骤中的各项参数值,在J的基础上引入w_{kj}的约束条件(见公式(1))构造优化函数J_1:

$$J_1(C,V,W)=\sum_{k=1}^{K}\left(\sum_{j=1}^{D}w_{kj}X_{kj}-\frac{\sum_{j=1}^{D}\sqrt{w_{kj}}-1}{D(\sqrt{D}-1)}\sum_{j=1}^{D}X_{kj}+\lambda_k\left(\sum_{j=1}^{D}w_{kj}-1\right)\right),$$

其中,λ_k(k=1,2,…,K)为拉格朗日乘子.

在‘M’步骤,C已给定,因此可以通过令$\frac{\partial J_1}{\partial v_{kj}}=0$($k$=1,2,…,$K$;$j$=1,2,…,$D$)来求解$V$.经过推导有

$$v_{kj}=\frac{1}{|C_k|}\sum_{\boldsymbol{x}_i\in C_k}x_{ij} \tag{4}$$

这里,$|C_k|$表示C_k的数据点数目.同理,令$\frac{\partial J_1}{\partial w_{kj}}=0$($k$=1,2,…,$K$;$j$=1,2,…,$D$)来推导$w_{kj}$的计算公式,经整理有

$$w_{kj}=\begin{cases}\dfrac{1}{4D^2(\sqrt{D}-1)^2}\left(\dfrac{1}{X_{kj}+\lambda_k}\right)^2\times\left(\sum_{l=1}^{D}X_{kl}\right)^2, & \text{if } \sum_{l=1}^{D}X_{kl}>0\\ \dfrac{1}{D}, & \text{otherwise}\end{cases} \tag{5}$$

公式(5)形式上与FWKM[9]和FSC[10]给出的维度权重计算式(见公式(2))较为接近.但是,FWKM和FSC的加权因子β和α作为算法的参数需要用户来设置.此外,为了避免除0错误而附加的δ和ε(形式上相当于λ_k)被定义为

经验设定的常数.这种设置必然无法反映数据集本身的特点,因而降低了算法的适应性.下节讨论确定λ_k的方法.

2.4 自适应参数设置

给定数据集的划分,求解每个划分的最优投影子空间可以看作是一个带等式约束的非线性规划问题,其目标函数为 J,约束条件为公式(1),J_1 通过引入拉格朗日乘子λ_k(k=1,2,…,K)将之转换为无约束的优化问题.根据优化理论[20],λ_k在(−∞,+∞)区间存在.但是,通过分析公式(5)可知,可行的λ_k取值范围是受限的.根据投影聚类的基本假设[8,13],w_{kj}的大小应与 X_{kj}成“反比”.为使公式(5)符合要求,有意义的λ_k取值范围应缩小为(−MIN_X_k,+∞).这里,MIN_X_k表示 X_{kj}(j=1,2,…,D)中的最小值.此外,λ_k在该区间还应该是唯一的.

根据 ASC 的算法过程,λ_k的求解在算法的‘M’步骤进行,此时,X_{kj}(j=1,2,…,D)已给定.那么,λ_k的存在性可描述为:给定 X_{kj}(j=1,2,…,D),是否存在唯一的λ_k>−MIN_X_k,使得公式(5)满足约束条件(1).为此,结合公式(5)和公式(1)得出λ_k的约束方程如下:

$$\psi(\lambda_k)=(\sum_{j=1}^{D}X_{kj})^2\sum_{j=1}^{D}\left(\frac{1}{X_{kj}+\lambda_k}\right)^2-4D^2(\sqrt{D}-1)^2=0 \tag{6}$$

定理 1 说明,存在唯一的λ_k>−MIN_X_k满足该方程.

定理 1. 存在唯一的λ_k>−MIN_X_k使得$\psi(\lambda_k)$=0.

证明:易验证λ_k>−MIN_X_k时$\psi(\lambda_k)$单调递减,再由 $\lim_{\lambda_k\to-\mathrm{MIN}\_X_k}\psi(\lambda_k)>0$ 和 $\lim_{\lambda_k\to+\infty}\psi(\lambda_k)<0$ 得到定理结论. □

根据算法过程和定理 1,ASC 算法自适应参数λ_k的值对应于方程$\psi(\lambda_k)$=0 的根.有多种算法[21]可用于求解方程(6),本文使用了牛顿法[21].ASC 这个计算过程的引入,在一定程度上增加了算法的时间开销,但是该过程使得算法的关键参数能够根据数据集的特点和每个算法步骤产生的聚类结果进行自适应调整.与相关研究[9,10]采用固定的经验值相比,这种代价换取了算法的高度自适应性,是值得的.

2.5 算法分析

ASC 算法是经典 k-means 算法[1]的扩展,主要区别在于增加了一个计算最佳子空间的步骤和重新定义了各迭代步骤使用的计算公式.其时间复杂度为 $O(KNDP)$.其中,P 是算法的迭代步数.以下分析 P 是有限的,也就是说,ASC 可以在有限的迭代步骤后收敛.根据 ASC 的终止条件,若 V 和 W 的变化小于ε,则可以认为 V 和 W 没有发生变化,而给定 V 和 W 由算法循环第 1 个步骤所产生的数据集划分 C 是确定的.这意味着,给定ε数据集,所有可能的划分组合都是有限的.那么,命题“ASC 可以在有限迭代步骤后收敛”与“任意一个 C 在 ASC 算法过程中最多只会出现 1 次”是等价的.下面简要证明后者的正确性.

令 p_1 和 p_2 表示两次迭代步骤的序号,p_1≠p_2.由 EM 的原理可知,$J_1(C^{(p_1)},V^{(p_1)},W^{(p_1)})\neq J_1(C^{(p_2)},V^{(p_2)},W^{(p_2)})$.使用反证法.假设 $C^{(p_1)}=C^{(p_2)}$,算法的‘M’步骤使用公式(4)来更新 V,而公式(4)与 W 无关,故若 $C^{(p_1)}=C^{(p_2)}$,则 $V^{(p_1)}=V^{(p_2)}$;之后,算法使用公式(5)更新 W,其值是由 C 和 V 确定的,故若 $C^{(p_1)}=C^{(p_2)}$ 且 $V^{(p_1)}=V^{(p_2)}$,则 $W^{(p_1)}=W^{(p_2)}$.这样,$J_1(C^{(p_1)},V^{(p_1)},W^{(p_1)})=J_1(C^{(p_2)},V^{(p_2)},W^{(p_2)})$,与前提条件矛盾,故假设不成立.也就是说,若 p_1≠p_2,则必有 $C^{(p_1)}\neq C^{(p_2)}$,说明 ASC 可以在有限的迭代步骤后收敛.

3 实验与分析

实验验证包括算法有效性和算法效率两个方面,并与若干相关工相比较.实验设备为配置 2.4GHz CPU 和 2GB RAM 的计算机.

3.1 实验设置

实验选择 PROCLUS[6],EWKM[8],FWKM[9]和 FSC[10]这 4 种聚类算法作为对比对象.PROCLUS[6]是第 1 种投影聚类算法,且本文采用了 PROCLUS 提出的方法用于生成合成数据(见第 3.2 节).EWKM[8]是新近提出来的软子空间聚类算法,它采用的指数加权方法具有代表性(使用这种加权方法的算法还包括 LAC[7]等);FWKM[9]和 FSC[10]采用多项式维度加权方法,与 ASC 算法比较接近.

与 ASC 算法相比,4 种对比算法都需要用户给定额外的(参数 K 除外,这是 5 种算法都需要的)算法参数.PROCLUS[6]需要参数 l 指定投影子空间的平均维度数,实验中 l 分别设定为 $D/2$ 和 $D/3$.这里,D 表示每个数据集实际的数据维度数.对于 EWKM[8],参数γ根据作者建议设定为 0.5.此外,为测试其对参数的敏感性,实验也将报告γ调整为 1.0 的结果.FWKM[9]和 FSC[10]的参数β和α分别根据文献[9,10]的建议,设定为 1.5 和 2.1.同样地,实验中还将β和α分别设定为 2.0 和 3.0 来验证对比算法的适应性.

各种算法的性能不仅在若干个实际应用数据集上进行测试,还将在合成数据集上进行实验对比.这是因为,合成数据可以控制数据集的簇结构,如簇的数目、相关维度的比例和簇的大小等,便于分析算法对各个数据集的适应性,以及算法性能与簇结构以上特性之间的关系.由于各种数据都已包含有正确的类标号,可以通过比较聚类结果与实际类标号之间的差异来衡量算法的聚类精度(这是一种外部聚类的有效性方法[17]).数据具有的这些类标号仅仅用于这个目的,在聚类过程中,类标号对各种算法都是未知的.实验采用 Micro-F1 和 Macro-F1 指标[14]来衡量各种算法的聚类精度.

3.2 合成数据实验结果

为比较和分析聚类算法对各种数据的适应性,根据文献[6]提供的方法合成了 4 个数据集,见表 1.表 1 的最后一列为各簇类包含的数据点数目.

Table 1 Summarized parameters of the synthetic datasets

表 1 合成数据集参数汇总表

DB	Size (N)	Dimension (D)	Average number of dimensions (l)	Number of clusters (K)	Size of each cluster
DS1	4 000	40	20	4	535:855:1154:1456
DS2	8 000	60	30	6	351:914:1213:1542:1806:2174
DS3	10 000	80	40	8	255:781:948:1467:1517:1672:1669:1691
DS4	20 000	100	50	10	225:714:1275:2382:2435:2447:2515:2525:2595:2887

由于 PROCLUS[6]使用抽样技术,算法的每一次执行都可能产生不一样的聚类结果.其他 4 种算法(包括 ASC 算法)都是类 k-means 的,从不同的初始簇中心出发,k-means 型算法也可能导致有差异的聚类结果.因此,对于同一个数据集,实验分别运行 5 种算法各 10 次,最后报告平均的聚类性能.为便于公平比较,除 PROCLUS 外的 4 种算法每次运行都从同样选择的初始簇中心开始.表 2 给出各种算法在 4 个合成数据集上取得的平均聚类精度.

Table 2 Comparison of the clustering results on the synthetics datasets

表 2 合成数据集上的聚类结果比较

	DS1		DS2		DS3		DS4	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
ASC	**0.981 7**	**0.968 8**	**0.990 7**	**0.961 1**	**0.975 2**	**0.927 5**	0.965 2	**0.917 6**
PROCLUS (l=D/2)	0.895 5	0.865 9	0.824 9	0.714 1	0.800 3	0.727 9	0.795 0	0.684 1
PROCLUS (l=D/3)	0.901 3	0.859 3	0.843 7	0.747 6	0.829 5	0.752 8	0.776 6	0.684 1
EWKM (γ=0.5)	0.668 1	0.628 9	0.439 7	0.406 0	0.410 2	0.364 6	0.366 9	0.324 5
EWKM (γ=1.0)	0.736 2	0.650 3	0.520 4	0.447 9	0.439 7	0.362 0	0.394 1	0.325 3
FWKM (β=1.5)	0.763 2	0.669 5	0.956 0	0.806 4	0.894 1	0.828 2	0.930 3	0.838 8
FWKM (β=2.0)	0.938 4	0.881 6	0.955 9	0.806 4	0.951 9	0.885 1	0.952 3	0.896 6
FSC (α=2.1)	0.968 5	0.940 0	0.956 1	0.806 5	0.974 4	0.925 6	0.952 2	0.915 8
FSC (α=3.0)	0.956 7	0.932 3	0.880 2	0.750 8	0.968 1	0.920 6	**0.967 7**	0.916 7

如表 2 所示,ASC 算法绝大多数情况下都获得了最高的聚类精度.尽管对比算法在一些数据集上也可以获得较好的性能,如 FSC 在 DS1 和 DS4 上的表现,但其性能与算法参数的设置密切相关.例如在 DS2 上,当α=2.1 时,FSC 算法的聚类精度达到 0.956 1(Micro-F1);当参数调整为 3.0 时,急剧下降为 0.880 2.从表 2 可以看出,PROCLUS,EWKM 和 FWKM 算法也存在类似现象.需要指出的是,一种算法所需的参数设置应与所处理的数据集的特性有关,包括数据维度数、每个簇类及所在的投影子空间的大小等.而这些算法都使用统一的经验设置,必然导致算法的适应性下降.作为对比,ASC 是一种自适应的算法,它在聚类过程中根据数据集的特性动态地确

定算法的参数,使得 ASC 能够适应不同的数据集,获得更高的聚类精度.

图 1 显示 5 种算法聚类 4 个数据集需要的平均运行时间(单位:s).与具有类似算法结构的 FWKM 和 FSC 相比, ASC 所需的算法时间略有增加.这是由于 ASC 在算法过程中增加了一个计算最佳投影子空间的优化过程,以换取算法对不同聚类问题的泛化能力.图 1 同时显示,PROCLUS 具有较高的效率.需要说明的是,PROCLUS 是一种硬子空间聚类算法.与 ASC,FWKM,FSC 算法相比,EWKM 明显地需要更多时间来完成聚类.实验中发现,为 EWKM 选择合适的算法参数较为困难,因该算法需要比其他算法更多的迭代步骤,增加了算法运行时间.

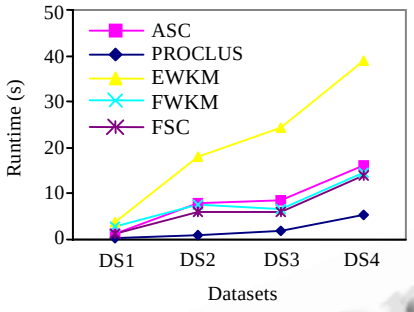

Fig.1 Comparison of the runtime of different algorithms

图 1 不同算法的运行时间对比

下面分析 ASC 算法对数据点数 N、数据维度数 D 和簇数目 K 这 3 个数据集参数的可伸缩性.实验数据集采用上述数据合成方法重新生成,当合成一个参数的可伸缩性测试数据时,其他两个参数采用固定的设置.图 2 显示 ASC 的运行时间与数据点数、数据维度数和簇数目之间的关系.在测试数据规模的可伸缩性时,合成参数为 K=4,D=20 和 l=8.如图 2(a)所示,ASC 的运行时间与数据点的数目 N 呈线性关系.在数据维度数和簇数目的可伸缩性测试上,数据集的合成参数分别设定为 N=20000,K=4,l=D×0.25 和 N=20000,D=20,l=8.如图 2(b)和图 2(c)所示,正如第 2.5 节所分析的那样,ASC 的运行时间与数据维度数 D 和簇数目 K 之间也呈现出线性关系.

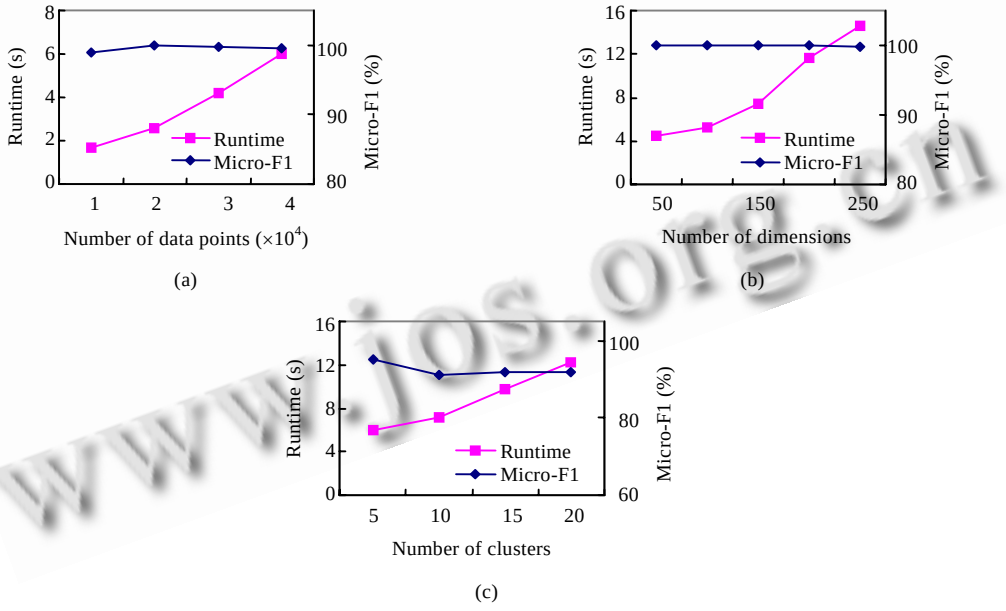

Fig.2 Relationships between the runtime of ASC, and different numbers of data points, dimensions and clusters

图 2 ASC 算法的运行时间与数据点数、数据维度数和簇数目之间的关系

3.3 实际数据实验结果

下面在4个常用的实际应用数据集上检验ASC算法,并与其他4种算法作对比.数据集的有关信息参见表3.第1个实际数据集是Iris,Iris可能是相关研究最频繁引用的一种数据.实验选择该数据的目的还在于Iris的数据维度只有4维,可以测试ASC算法对较低维度数据的适应性.如表3所示,其他3个数据集的数据维度数逐渐递增,分别用于测试ASC算法对具有几十、几百和几千个维度的实际数据集的有效性.

第2个数据集是SpamBase,它与Iris一样都来自UCI machine learning repository(ftp.ics.uci.edu:pub/machine-learning-databases),也是一个机器学习、模式识别领域相关研究常用的测试数据集,用于垃圾邮件过滤研究.另外两个数据集取自常用的分类语料库.Classic4取自著名的cluto clustering工具箱(http://www.users.cs.umn.edu/~karypis/cluto),包括CACM,CISI,CRAN和MED这4类期刊论文文档.Email-1431是基于内容的垃圾邮件过滤研究中常用的一个英文电子邮件语料库[7],经词干分析等预处理之后,选取约2 000个出现频度最高的单词作为文档的特征.Classic4和Email-1431采用最简单的VSM(vector space model)模型[2]表示,文档向量的每个元素仅对应于词在文档中出现的频度.SpamBase,Classic4和Email-1431都是文档数据,故在预处理过程中,每个向量$\boldsymbol{x}$都事先变换成单位长度,即$\|\boldsymbol{x}\|_2=1$.

Table 3 Some information about the real-world datasets

表3 实际应用数据集的有关信息

DB	Size (*N*)	Dimension (*D*)	Number of the clusters (*K*)	Size of each cluster
Iris	150	4	3	50:50:50
SpamBase	460 1	54	2	1813:2788
Classic4	709 4	800	4	1033:1398:1460:3203
Email-1431	143 1	200 2	2	642:789

表4给出了各种算法在4个实际应用数据集上的聚类结果,所列数据为每种算法对数据集进行100次聚类获得的平均的Micro-F1和Macro-F1指标值.由表4可见,ASC算法在4个数据集上均获得了较高的聚类精度,尤其是在只有4维的低维数据Iris和具有高达2002维的Email-1431上,这进一步验证了ASC算法对特性各异的数据集的适应性.4种对比算法在参数设置合适的情况下也可以取得较好的结果,但当调整参数设置时,其聚类精度也随之发生变化.而ASC算法不需要参数选择,可以通过自适应的算法过程获得较好的聚类结果.

Table 4 Comparison of the clustering results on the real-world datasets

表4 实际应用数据集上的聚类结果比较

	Iris		SpamBase		Classic4		Email-1431	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
ASC	**0.925 7**	**0.924 7**	**0.791 7**	**0.791 0**	**0.809 9**	**0.830 2**	**0.930 0**	**0.929 8**
PROCLUS (l=D/2)	0.852 4	0.845 8	0.603 5	0.570 9	0.634 1	0.500 2	0.612 6	0.558 2
PROCLUS (l=D/3)	0.850 3	0.844 6	0.559 9	0.465 6	0.546 4	0.331 8	0.573 4	0.500 8
EWKM (γ=0.5)	0.912 2	0.910 8	0.587 9	0.427 9	0.490 5	0.275 3	0.597 3	0.475 2
EWKM (γ=1.0)	0.908 2	0.906 9	0.567 3	0.415 6	0.575 7	0.404 2	0.605 1	0.487 4
FWKM (β=1.5)	**0.925 7**	**0.924 7**	0.633 3	0.490 1	0.601 3	0.450 0	0.598 0	0.477 8
FWKM (β=2.0)	0.925 2	0.924 1	0.609 8	0.519 3	0.631 5	0.514 3	0.733 5	0.672 4
FSC (α=2.1)	0.925 2	0.924 1	0.618 6	0.481 4	0.620 0	0.497 1	0.683 9	0.622 3
FSC (α=3.0)	0.910 9	0.909 9	0.751 0	0.743 4	0.627 9	0.539 0	0.816 4	0.785 1

ASC算法由于其自适应的特点,可以获得比其他算法更为稳定的聚类结果.以数据维度数最高的Email-1431为例,鉴于k-means型算法的局限性[1],算法容易陷入局部最优,使得聚类结果因选择了不同的初始簇中心而反差很大.对于高维数据而言,目前尚缺乏有效的初始簇中心选择方法.图3给出了Email-1431数据集上各种算法分别进行100次聚类的统计结果,其横坐标是以Micro-F1衡量的聚类精度落在相应区间的聚类次数.PROCLUS,EWKM,FWKM和FSC算法的参数分别为l=D/2,γ=0.5,β=1.5和α=2.1.如图3所示,从100组不同的初始状态出发,ASC算法近80次的聚类结果的精度达到0.9以上,而4种对比算法获得0.9以上精度的聚类过程均不超过20次,低于总数的20%.图3显示,对比算法的聚类结果对初始状态是敏感的.例如,FWKM和EWKM算法在80%的聚类过程中只能获得小于0.7的精度,ASC算法则较为鲁棒,其聚类精度稳定在0.8以上,其中,约

80%的聚类过程可以获得 0.9 以上的精度.

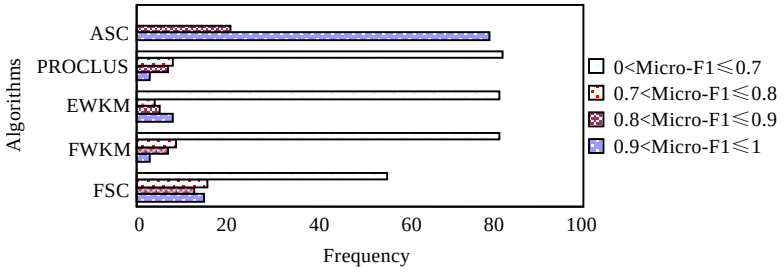

Fig.3 Comparison of the clustering results across 100 different executions

图 3 算法 100 次聚类结果对比

ASC 算法的性能优势来源于它的自适应特性.根据 ASC 算法过程,在每一个迭代步骤生成数据集的划分后(包括从初始选择的簇中心出发生成的初始划分),ASC 算法能够根据数据集的特点自动地进行参数调整,更准确地估计出每个划分所在的最佳投影子空间,从而可以在一定程度上避免算法过早地陷于局部最优,获得较为稳定的聚类结果.

4 小结及工作方向

本文提出一种自适应的算法 ASC,用于高维数据的软子空间聚类分析.ASC 使用新提出的聚类优化目标函数,在最小化子空间簇类的簇内紧凑度的同时,最大化每个簇类的投影子空间.由此推导了新的局部特征加权计算公式和自适应参数的约束方程,并证明了方程根的存在性.与现有的其他软子空间聚类算法相比,ASC 不需要用户给定额外的算法参数,而是在聚类过程中根据数据集及其划分的信息,动态地确定参数的最优取值,以确定簇类所在的最佳投影子空间.在实际应用数据和合成数据上的实验结果表明,ASC 算法可以有效地进行软子空间聚类,其聚类精度优于新近提出的有关算法,并大幅度提升了聚类结果的稳定性.下一步的工作重点是找出求解 λ_k 约束方程更简便的方法及设计针对高维数据稀疏聚类的初始簇中心选择方法,以进一步提高算法的稳定性.

致谢 在此,我们感谢对本文提供了有价值评论的匿名审稿人.

References:

[1] Berkhin P. Survey of clustering data mining techniques. In: Kogan J, Nicholas C, Teboulle M, eds. Grouping Multidimensional Data: Recent Advances in Clustering. Berlin: Springer-Verlag, 2006. 25−71.

[2] Leopold E, Kindermann J. Text categorization with support vector machines: How to represent texts in input space? Machine Learning, 2002,46(1-3):423−444. [doi: 10.1023/A:1012491419635]

[3] Verleysen M. Learning high-dimensional data. In: Ablameyko S, Goras L, Gori M, Piuri V, eds. Proc. of the Limitations and Future Trends in Neural Computation. Siena: IOS Press, 2003. 141−162.

[4] Parsons L, Haque E, Liu H. Subspace clustering for high dimensional data: A review. ACM SIGKDD Explorations Newsletter, 2004,6(1):90−105. [doi: 10.1145/1007730.1007731]

[5] Yang Q, Wu X. 10 challenging problems in data mining research. Int'l Journal of Information Technology and Decision Making, 2006,5(4):597−604. [doi: 10.1142/S0219622006002258]

[6] Aggarwal CC, Procopiuc C, Wolf JL, Yu PS, Park JS. Fast algorithm for projected clustering. In: Delis A, Faloutsos C, Ghandeharizadeheds S, eds. Proc. of the ACM-SIGMOD. New York: ACM Press, 1999. 61−71.

[7] Domeniconi C, Gunopulos D, Ma S, Yan B, Al-Razgan M, Papadopoulos D. Locally adaptive metrics for clustering high dimensional data. Data Mining and Knowledge Discovery, 2007,14(1):63−97. [doi: 10.1007/s10618-006-0060-8]

[8] Jing L, Ng MK, Huang JZ. An entropy weighting *k*-means algorithm for subspace clustering of high-dimensinoal sparese data. IEEE Trans. on Knowledge and Data Engineering, 2007,19(8):1−16. [doi: 10.1109/TKDE.2007.250581]

[9] Huang JZ, Ng MK, Rong H, Li Z. Automated variable weighting in *k*-means type clustering. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2005,27(5):657−668.

[10] Gao G, Wu J, Yang Z. A fuzzy subspace clustering algorithm for clustering high dimensional data. In: Li X, Zaiane OR, Li Z, eds. Proc. of the ADMA. Berlin, Heidelberg: Springer-Verlag, 2006. 271−278.

[11] Gao J, Wang S. Fuzzy clustering algorithm with ranking features and identifying noise simultaneously. Acta Automatica Sinica, 2009,35(2):145−153 (in Chinese with English abstract). [doi: 10.3724/SP.J.1004.2009.00145]

[12] Woo KG, Lee JH, Kim MH, Lee YJ. FINDIT: A fast and intelligent subspace clustering algorithm using dimension voting. Information and Software Technology, 2004,46(4):255−271. [doi: 10.1016/j.infsof.2003.07.003]

[13] Moise G, Sander J, Ester M. Robust projected clustering. Knowledge Information System, 2008,14(3):273−298. [doi: 10.1007/s10115-007-0090-6]

[14] Chen L, Jiang Q, Wang S. A probability model for projective clustering on high dimensional data. In: Giannotti F, Gunopulos D, Turini F, Zaniolo C, Ramakrishnan N, Wu X, eds, Proc. of the IEEE ICDM. Los Alamitos: IEEE Computer Society Press, 2008. 755−760.

[15] Bouguessa M, Wang S, Sun H. An objective approach to cluster validation. Pattern Recognition Letters, 2006,27(13):1419−1430. [doi: 10.1016/j.patrec.2006.01.015]

[16] Procopiuc CM, Jones M, Agarwal PK, Murali TM. A monte carlo algorithm for fast projective clustering. In: Michael JF, Bongki M, Anastassia A, eds. Proc. of the ACM-SIGMOD. New York: ACM Press, 2002. 418−427.

[17] Patrikainen M, Meila M. Comparing subspace clusterings. IEEE Trans. on Knowledge and Data Engineering, 2006,18(7):902−916. [doi: 10.1109/TKDE.2006.106]

[18] Aggarwal CC, Yu PS. Finding generalized projected clusters in high dimensional spaces. In: Chen W, Naughton JF, Bernstein PA, eds. Proc. of the ACM-SIGMOD. New York: ACM Press, 2000. 70−81.

[19] Xu L, Jordan MI. On convergence properties of the EM algorithm for Gaussian mixtures. Neural Computation, 1996,8(1):129−151. [doi: 10.1162/neco.1996.8.1.129]

[20] Xue Y. Optimization Theory and Method. Beijing: Beijing University of Technology Press, 2001 (in Chinese).

[21] Press WH, Teukolsky SA, Vetterling WT, Flannery BP. Numerical Recipes in C++: The Art of Scientific Computing. 2nd ed., Cambridge: Cambridge University Press, 2002.

附中文参考文献:

[11] 皋军,王士同.具有特征排序功能的鲁棒性模糊聚类方法.自动化学报,2009,35(2):145−153. [doi: 10.3724/SP.J.1004.2009.00145]

[20] 薛毅.最优化原理与方法.北京:北京工业大学出版社,2001.

陈黎飞(1972－),男,福建长乐人,博士,讲师,主要研究领域为数据挖掘,模式识别.

姜青山(1962－),男,博士,教授,博士生导师,主要研究领域为数据挖掘,图像处理,数据库系统,模糊集理论与应用.

郭躬德(1965－),男,博士,教授,博士生导师,主要研究领域为人工智能,数据挖掘.