计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2022,31(12):266−272 [doi: 10.15888/j.cnki.csa.008857]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于熵的平衡子空间 K-means 算法①

康泰榕, 何振峰

(福州大学 计算机与大数据学院, 福州 350108)
通信作者: 康泰榕, E-mail: 540338741@qq.com

摘　要: 在许多数据挖掘的实际应用中要求每一个类别的实例数量相对平衡. 而独立子空间聚类的熵加权 K-means 算法 (EWKM) 会产生不均衡的划分, 聚类质量很差. 本文定义了一种兼顾平衡划分与特征分布的多目标熵, 然后应用该熵改进了 EWKM 算法的目标函数, 同利用迭代方法和交替方向乘子法设计其求解流程, 并提出基于熵的平衡子空间 K-means 算法 (EBSKM). 最后, 在 UCI、UCR 等公开数据集进行聚类实验, 结果表明所提算法在准确率和平衡性方面都优于同类算法.

关键词: 子空间聚类; 平衡聚类; 特征加权; K-means

引用格式: 康泰榕,何振峰.基于熵的平衡子空间 K-means 算法.计算机系统应用,2022,31(12):266–272. http://www.c-s-a.org.cn/1003-3254/8857.html

Entropy-based Balanced Subspace K-means Algorithm

KANG Tai-Rong, HE Zhen-Feng

(College of Computer and Data Science, Fuzhou University, Fuzhou 350108, China)

Abstract: In many practical applications of data mining, instances for each cluster are often required to be balanced in number. However, the entropy-weighted K-means algorithm (EWKM) for independent subspace clustering leads to unbalanced partitioning and poor clustering quality. Therefore, this study defines a multi-objective entropy that takes balanced partitioning and feature distribution into account and then employs the entropy to improve the objective function of the EWKM algorithm. Furthermore, the study designs the solution process by using the iterative method and alternating direction method of multipliers and proposes the entropy-based balanced subspace K-means algorithm (EBSKM). Finally, the clustering experiments are conducted in public datasets such as UCI and UCR, and the results show that the proposed algorithm outperforms similar algorithms in terms of accuracy and balance.

Key words: subspace clustering; balanced clustering; feature weighting; K-means

聚类是一项重要的数据分析技术, 子空间聚类是传统聚类算法的扩展, 试图寻找数据集在不同的特征子集中存在的簇[1]. 对于给定的数据集, 子空间聚类一般可以找出多种不同的划分方案, 可以利用数据集本身的属性信息或用户提供的有效信息, 来寻找特定需求的子空间划分方案, 并得到对应条件的聚类结果. 例如, 对于 DNA 序列可以使用子空间聚类算法来识别序列中的隐藏模式 (簇), 更重要的是通过无监督的方式执行自动变量选择来揭示感兴趣的潜在生物学概念[2].

子空间聚类的熵加权 K-means 算法 (EWKM) 是由 Jing 等人[3] 扩展了加权 K-means 算法引入了权重熵, 从而使得更多维度加入到聚类过程中, 该算法提出后就得到了广泛应用. 目前, 有许多算法都是基于该算法框架利用不同信息或从不同角度进行扩展, 如利用簇间信息来扩展, Xiong 等人提出 ERKM 算法[4] 在统一的子空间中, 通过最大化簇中心与不属于该簇的点

① 基金项目: 福建省自然科学基金 (2018J01794)
收稿时间: 2022-04-10; 修改时间: 2022-05-09; 采用时间: 2022-05-28; csa 在线出版时间: 2022-08-12

之间的距离, 使实例在子空间中保持簇内距离最小, 各个簇之间距离最大的状态. 利用数据集的特征信息来扩展, 例如 Yang 等人提出的 FRFCM 算法[5], 在 FCM 模型中引入特征权重熵, 同时利用特征的均值-方差比来评估每个特征的重要程度, 利用较小的权值来消除不相关特征给聚类过程带来的干扰. 也有通过修改权重熵的角度来改进, Huang 等人[6] 利用l^2-norm来代替权重熵项以此提高特征选择的有效性, 让更多的特征参与到子空间聚类当中.

上述聚类方法均有应用 EWKM 算法中的特征加权方法, 并在其基础上进行扩展改进, 已成功应用到很多领域. 但是, 上述聚类算法均没考虑在实例分配过程中各个簇的实例数量是否均衡, 所以无法很好解决平衡聚类问题. 同时, 利用平衡聚类也可以帮助子空间生成保持平衡结构的特征权重. 在许多实际应用中, 用户是希望聚类结果能有较好的平衡性, 应用需求也是研究平衡聚类的一个动机. 例如在无线传感网络中[7,8], 理想情况下应当使每个网络节点的样本数量大致相同, 否则会增加能源消耗; 在分布式数据管理中, 为了提高分布式数据管理系统在查询时节点间数据传输的性能, 通常需要尽可能平衡地分配每个节点的数据量, 避免出现负载较大的单个节点[9]; 在营销活动中, 为保证公平和效率需要将客户平衡划分多个集群, 以保证每个销售人员工作量相同. 除了满足实际需求外, 平衡划分往往会降低初始化的敏感性, 因此即使在不需要平衡的情况下, 也会产生有益的效果[10,11].

平衡聚类算法能生成各簇实例数量大致相同的聚类结果, 可以分为软平衡聚类与硬平衡聚类, 前者强调平衡只是一个目标, 而不是强制性的要求; 后者强调聚类大小应当被严格设置为一个固定数目. 现有软平衡聚类的方法有: 文献 [12] 将算法分成 3 个部分, 先利用重复事件与更新理论对数据集进行抽样获得代表性较高的抽样集, 再利用被平衡约束修改后的 K-means 算法对抽样集聚类, 最后将那些未被抽样的点利用稳定婚姻匹配算法逐个分配到合适的簇中. 文献 [13] 利用实例标签信息构造标签分布熵来评价聚类的平衡度, 然后将该熵、模糊隶属度矩阵与标签矩阵之间的平方损失引入 FCM 模型中. 硬平衡聚类的方法有文献 [14] 在最小平方和聚类中对每一个簇大小设置约束, 同时利用可变领域启发式方法来寻找最优划分. 文献 [15] 采用线性规划的方法把每个簇大小作为约束条件, 从而完成平衡聚类.

本文利用实例分布信息扩展 EWKM 算法, 定义了一种兼顾平衡划分与特征分布的多目标熵, 同时应用该熵提出基于熵的平衡子空间 K-means 算法, 来解决子空间聚类算法中聚类结果不均衡的问题.

1 熵加权 K-means 算法

对于给定数据集$X=[X_1,X_2,\cdots,X_n]\in R^{n\times D}$, 其中$n$为实例集中的实例数量, D表示实例的维度. 假设数据集X划分为k类, EWKM 算法模型为:

$$\begin{cases}\min\limits_{G,F,W}\sum\limits_{j=1}^{k}\sum\limits_{i=1}^{n}F_{ij}\sum\limits_{d=1}^{D}W_{jd}\left\|X_{id}-G_{jd}\right\|^2-\tau H_F\\ \text{s.t.}\begin{cases}F_{ij}\in\{0,1\}^{n\times k},\sum\limits_{j=1}^{k}F_{ij}=1\\ H_F=-\sum\limits_{j=1}^{k}\sum\limits_{d=1}^{D}W_{jd}\log\left(W_{jd}\right)\\ 0\leqslant W_{jd}\leqslant 1,\sum\limits_{d=1}^{D}W_{jd}=1\end{cases}\end{cases}\tag{1}$$

其中, 超参数τ主要用于激励在更多维度上聚类. $F=\{F_{ij}\}\in R^{n\times k}$为 0–1 实例隶属矩阵, $F_{ij}=1$, 表示X_i属于第j簇. $W\in R^{k\times D}$为特征权重矩阵, $W_j=\left(W_{j1},W_{j2},\cdots,W_{jD}\right)$为第$j$个簇中不同维度的权重, 权值越大则表示此维度对于该簇的贡献也越大. G表示聚类中心矩阵. 式 (1) 中第 1 项表示簇内距离之和, 第 2 项是权重熵的和, 目的是避免算法通过极个别维度来识别簇.

算法通过迭代的方式来求解模型, 即固定某两个参数值来求解另一个参数. 聚类中心矩阵G、特征权重矩阵W和实例隶属矩阵F的更新公式如下所示.

在更新聚类中心矩阵G时, 固定参数F, W得:

$$G_{jt}=\frac{\sum\limits_{i=1}^{n}F_{ij}X_{it}}{\sum\limits_{i=1}^{n}F_{ij}},\ 1\leqslant j\leqslant k,\ 1\leqslant t\leqslant D\tag{2}$$

更新特征权重矩阵W, 固定参数G, F得:

$$W_{jt}=\frac{\exp\left(\frac{-Dist_{jt}}{\tau}\right)}{\sum\limits_{d=1}^{D}\exp\left(\frac{-Dist_{jd}}{\tau}\right)}\tag{3}$$

其中, $Dist_{jd}=\sum\limits_{i=1}^{n}F_{ij}\left\|X_{id}-G_{jd}\right\|^2$, 表示在维度$d$上, 属于

第j个簇的实例到该簇中心的距离平方和.

更新实例隶属矩阵F, 固定参数G, W得:

$$F_{ij}=\begin{cases}1, & \text{if } \sum_{d=1}^{D}W_{jd}\|X_{id}-G_{jd}\|^2\leqslant\sum_{d=1}^{D}W_{td}\|X_{id}-G_{td}\|^2,\\ & 1\leqslant t\leqslant k\\ 0, & \text{otherwise}\end{cases} \tag{4}$$

这里给出 EWKM 算法的具体步骤如算法 1.

算法 1. EWKM 算法

输入: 数据集X, 超参数τ, 聚类簇数k

输出: 实例隶属矩阵F

1) 初始化: 随机选择k个簇中心点, 组成中心点矩阵G, 并依据维度数量平均分配每一个簇的初始维度权重向量W_{j_v}.

2) REPEAT

3) 更新实例隶属矩阵F, 固定G, W利用式 (4).

4) 更新聚类中心矩阵G, 固定W, F利用式 (2).

5) 更新特征权重矩阵W, 固定F, G利用式 (3).

6) UNTIL 目标函数 (1) 前后两次计算结果不再发生变化.

EWKM 算法通过优化模型 (1) 同时刺激更多特征参与聚类, 避免只利用少量甚至单一特征识别簇的问题. 但是, EWKM 算法在其优化过程中没有考虑各个簇实例的分布情况, 不适用于平衡聚类的场景.

2 基于熵的平衡子空间 K-means 算法

针对 EWKM 算法中实例分布不均的问题, 本文考虑了聚类过程中常用的平衡约束, 利用平衡分布熵与特征权重熵定义了一个多目标熵, 并在 EWKM 算法基础上建立了结合该熵的一个新的目标函数, 提出基于熵的平衡子空间 K-means 算法 (entropy-based balanced subspace K-means algorithm (EBSKM), 并给出其求解过程.

2.1 适用于平衡子空间的多目标熵

设p_j为隶属第j个簇的实例个数占全体实例个数的比值可以用矩阵F表示为:

$$p_j=\frac{\sum_{i=1}^{n}F_{ij}}{n},\ 1\leqslant j\leqslant k \tag{5}$$

平衡分布熵公式如式 (6) 所示:

$$H_B(p)=-\sum_{j=1}^{k}p_j\log\left(p_j\right) \tag{6}$$

当$H_B(p)$取到最小时, 则所有簇的p_j数值相等, 此时的聚类结果中各簇实例分布最为平衡. 当$H_B(p)$越大, 则意味着划分越不平衡. 在文献 [16] 中将该熵作为约束项来保证划分平衡.

平衡子空间的多目标熵定义如式 (7):

$$H=\frac{\alpha}{\log(k)}H_B+\frac{1-\alpha}{k\log(D)}H_F \tag{7}$$

其中, α为权重系数, $\alpha\in[0,1]$, 当$\alpha=0$时, 退化为特征加权熵, 当$\alpha=1$时, 退化为平衡分布熵. α主要起到控制前后两项熵的重要程度. $1/\log(k)$, $1/k\log(D)$主要是用于平衡这两个熵的影响.

2.2 EBSKM 算法模型

为了使得 EWKM 算法能适用于平衡聚类, 我们在其基础上应用上述的多目标熵, 提出 EBSKM 的目标函数如式 (8) 所示:

$$\begin{cases}\min\limits_{G,F,W,p}\sum\limits_{j=1}^{k}\sum\limits_{i=1}^{n}F_{ij}\sum\limits_{d=1}^{D}W_{jd}\left\|X_{id}-G_{jd}\right\|^2-\gamma H\\ \text{s.t.}\begin{cases}F_{ij}\in\{0,1\}^{n\times k},\sum\limits_{j=1}^{k}F_{ij}=1\\ 0\leqslant W_{jd}\leqslant 1,\sum\limits_{d=1}^{D}W_{jd}=1\end{cases}\end{cases} \tag{8}$$

γ为超参数. 传统的子空间聚类算法往往侧重于对最具有鉴别能力的特征赋予较高权重, 而忽略了平衡性. EBSKM 算法通过引入多目标熵来解决传统算法中平衡性的问题, 同时自适应的特征加权也可以用于分析不同特征对聚类平衡性的贡献程度.

2.3 EBSKM 模型求解

在 EBSKM 算法模型中, 需要求解 4 个变量, 采取迭代方式来优化.

首先固定W,F,p求解聚类中心G. 聚类中心可以利用实例隶属矩阵F, 计算不同簇所在实例的平均值作为其聚类中心, 更新公式同式 (2).

同样的, 固定F,G,p求解W. 被固定的变量可视为常量, 所以可以消去部分含参项. 当$0\leqslant\alpha<1$时, 将目标函数 (8) 转化为:

$$\begin{cases}\min\limits_{W}\sum\limits_{d=1}^{D}W_{jd}Dist_{jd}+\dfrac{\gamma(1-\alpha)}{k\log(D)}\sum\limits_{d=1}^{D}W_{jd}\log\left(W_{jd}\right)\\ \text{s.t. } 0\leqslant W_{jd}\leqslant 1,\sum\limits_{d=1}^{D}W_{jd}=1\end{cases} \tag{9}$$

由于存在约束条件$\sum_{d=1}^{D} W_{jd}=1$, 构造 (9) 的拉格朗日函数如下:

$$L(W,\alpha)=\sum_{d=1}^{D} W_{jd}Dist_{jd}+\frac{\gamma(1-\alpha)}{k\log(D)}\sum_{d=1}^{D} W_{jd}\log\left(W_{jd}\right)-\alpha\left(\sum_{d=1}^{D} W_{jd}-1\right) \tag{10}$$

其中, α为拉格朗日乘子, 对W和α分别求偏导令其为零, 得到如下方程:

$$\frac{\partial L(W,\alpha)}{\partial\alpha}=\sum_{d=1}^{D} W_{jd}-1=0 \tag{11}$$

$$\frac{\partial L(W,\alpha)}{\partial W_{jt}}=Dist_{jt}+\frac{\gamma(1-\alpha)}{k\log(D)}\left(1+\log\left(W_{jt}\right)\right)-\alpha=0 \tag{12}$$

求解式 (11)、式 (12) 得特征权重更新公式:

$$W_{jt}=\frac{\exp\left(\frac{-k\log(D)Dist_{jt}}{\gamma(1-\alpha)}\right)}{\sum_{d=1}^{D}\exp\left(\frac{-k\log(D)Dist_{jd}}{\gamma(1-\alpha)}\right)} \tag{13}$$

当$\alpha=1$时, H只有平衡分布熵起作用.

在求解实例隶属矩阵F以及平衡分布向量p时, 利用交替方向乘子法 (ADMM) 来同时优化这两个参数, 通过引入拉格朗日乘子λ_j $(j=1,\cdots,k)$, 同时固定其他变量, 得到目标函数 (8) 的增广拉格朗日方程:

$$\begin{cases} L_\mu(F,p,\lambda)=\sum_{j=1}^{k}\sum_{i=1}^{n} F_{ij}\sum_{d=1}^{D} W_{jd}\left\|X_{id}-G_{jd}\right\|^2 \\ \quad +\frac{\gamma\alpha}{\log(k)}\sum_{j=1}^{k} p_j\log\left(p_j\right)+\sum_{j=1}^{k}\lambda_j\left(p_j-\frac{\sum_{i=1}^{n} F_{ij}}{n}\right) \\ \quad +\frac{\mu}{2}\sum_{j=1}^{k}\left(p_j-\frac{\sum_{i=1}^{n} F_{ij}}{n}\right)^2 \\ \text{s.t. } F_{ij}\in\{0,1\}^{n\times k},\sum_{j=1}^{k} F_{ij}=1 \end{cases} \tag{14}$$

其中, $\mu>0$为惩罚参数. ADMM 算法优化参数的迭代更新式如下:

$$p^{(t+1)}=\arg\min_{p} L_\mu\left(F^{(t)},p,\lambda^{(t)}\right) \tag{15}$$

$$F^{(t+1)}=\arg\min_{F} L_\mu\left(F,p^{(t+1)},\lambda^{(t)}\right) \tag{16}$$

更新拉格朗日乘子:

$$\begin{cases} \lambda_j^{(t+1)}=\lambda_j^{(t)}+\mu^{(t)}\left(p_j^{(t+1)}-\frac{\sum_{i=1}^{n} F_{ij}^{(t+1)}}{n}\right),1\leqslant j\leqslant k \\ \mu^{(t+1)}=\rho\mu^{(t)} \end{cases} \tag{17}$$

其中, $\rho>1$是一个给定的参数.

求解向量p, 利用式 (15) 可将向量p分为k个子问题, 其中第j项为:

$$\min_{p_j}\frac{\gamma\alpha}{\log(k)}p_j\log p_j+\lambda_j p_j+\frac{\mu}{2}{p_j}^2-\mu p_j\beta_j \tag{18}$$

其中, $\beta_j=\sum_{i=1}^{n} F_{ij}/n<1$. 对式 (18) 求导令其为 0 得求解$p_j$公式如式 (19) 所示:

$$\frac{\gamma\alpha}{\log(k)}\log p_j+\mu p_j+\lambda_j+\frac{\gamma\alpha}{\log(k)}-\mu\beta_j=0 \tag{19}$$

求解F时, 借鉴文献 [16] 的思想, 采用逐行求解的方式, 即对每一个实例求出当前状态下的最优分配方案. 由式 (16) 得$F_{i\cdot}$求解公式:

$$\begin{cases} \min_{F_{i\cdot}}\sum_{j=1}^{k}\sum_{i=1}^{n} F_{ij}\sum_{d=1}^{D} W_{jd}\left\|X_{id}-G_{jd}\right\|^2 \\ \quad -\sum_{j=1}^{k}\frac{\lambda_j F_{ij}+\mu p_j F_{ij}}{n}+\sum_{j=1}^{k}\frac{\mu {F_{ij}}^2+\mu F_{ij}\sum_{i\neq j} F_{ij}}{2n^2} \\ \text{s.t. } F_{ij}\in\{0,1\}^{n\times k},\sum_{j=1}^{k} F_{ij}=1 \end{cases} \tag{20}$$

从约束条件可以看出, 每一个实例只允许被分配到一个簇中. 因此, 在求解$F_{i\cdot}$时, 会有k个候选项, 每个候选项代表实例 i 属于第 j 簇的情况. 然后把这k个候选项分别带入式 (20) 中, 选择函数值最小的那个候选项作为$F_{i\cdot}$的解. 之后不断循环这一过程直到F不再变化, 即可得到完整的实例隶属矩阵F. 在优化F, p时, 很难同时对这两个变量进行优化, 因此利用 ADMM 算法对这两个变量进行分块优化, 其优势在于将大的全局问题分解为多个较小、较容易求解的局部子问题, 并通过协调子问题的解而得到大的全局问题的解, 以此来降低其优化难度.

算法 2 ADMM 算法求解F, p的具体流程.

算法 2. ADMM 算法求解 F, p

输入: X, W, G, 超参数 γ, α, 聚类簇数 k
输出: F, p

1) 初始化: μ, ρ
2) REPEAT
3) 更新 p 利用式 (19).
4) 更新 F 利用式 (20).
5) 更新拉格朗日乘子, 利用式 (17).
6) UNTIL 算法收敛

算法 2 中μ初始化值为 1, ρ初始化值为 1.1.

综合以上, EBSKM 算法求解的具体流程如算法 3.

算法 3. EBSKM 算法

输入: 数据集 X, 超参数 γ, α, 聚类簇数 k
输出: 实例隶属矩阵 F

1) 初始化: $[G, F] = kmeans(X, k)$
2) REPEAT
3) 更新 W, 固定 F, G, p 利用式 (13).
4) 更新 G, 固定 W, F, p 利用式 (2).
5) 更新 F, p, 利用算法 2.
8) UNTIL F 不再变化停止.

3 实验分析

为了验证 EBSKM 算法的有效性, 本文实验采用 4 个 UCI 数据集、10 个 UCR 数据集以及 1 个公开的图像数据集, 用于比较 EBSKM、K-means、EWKM、W-K-means[17]、OMBC[15]、FSBC[16] 的聚类性能.

3.1 数据集及预处理

Iris、Wine、Seeds、Digit 来自 UCI, Trace、Plane、SmoothSubspace (SMS)、Face (four)、Symbols、FacesUCR、ElectricDevices、HandOutlines、Mallat、StarLightCurves 来自 UCR, Jaffe 是来自公开的图像数据集. 因为本文所针对的实际应用场景是平衡聚类问题, 即要求所有的簇的实例个数要相同, 所以本文对每一组数据集中所有类别的实例个数随机删减至相同. 例如 Wine 数据集原来的样本总量为 178, 每个簇的实例数量为 59、71、48, 对大于最小规模的簇随机选择实例, 并删减至每个簇大小为 48. 每组数据集都采用 Z-Score 标准化, 实验具体数据如表 1 所示.

3.2 评价指标

本文通过利用标准互信息 (*NMI*) 来评价所有聚类算法的聚类性能, 利用标准熵 (*NE*) 来评价聚类结果的平衡性.

NMI 定义如下:

$$MI(X,Y)=\sum_{x_i\in X,y_j\in Y}p(x_i,y_j)\log\left(\frac{p(x_i,y_j)}{p(x_i)p(y_j)}\right) \tag{21}$$

$$NMI(X,Y)=2\frac{MI(X,Y)}{H(X)+H(Y)} \tag{22}$$

其中, $H(X)$, $H(Y)$ 表示类集 X, Y 的熵, $p(\cdot)$表示概率. *NMI* 的值越大说明算法的聚类性能越好.

NE 定义如下:

$$NE=-\frac{1}{\log(k)}\sum_{i=1}^{k}p_i\log(p_i) \tag{23}$$

NE=1 时表明聚类结果完全平衡, *NE*=0 表示聚类结果完全不平衡. 例如一个含有 3 个簇的数据集, 其聚类实例分布数量比例为 1/3, 1/2, 1/6, 可以认为该结果是不平衡的, 此时的 *NE* 为 0.9206.

表 1 数据集相关信息

数据集	样本规模 (原规模)	属性	类别数
Iris	150 (150)	4	3
Wine	144 (178)	13	3
Seeds	210 (210)	7	3
Trace	200 (200)	275	4
Plane	210 (210)	144	7
SMS	300 (300)	15	3
Digit	2000 (2000)	216	10
Face (four)	88 (112)	350	4
Symbols	972 (1020)	398	6
FacesUCR	672 (2250)	131	14
Jaffe	200 (213)	676	10
ElectricDevices	8769 (16637)	96	7
HandOutlines	990 (1370)	2709	2
Mallat	2400 (2400)	1024	8
StarLightCurves	3987 (9236)	1024	3

3.3 实验结果

本文提出的算法在实验中设置参数α为 0.6, γ为 200, 对于其他比较方法, EWKM$\gamma=0.5$, W-K-means $\beta=8$, FSBC$\lambda=n^2\times10^2, \tau=10^2$. 上述算法参数设置都是基于其论文中所给建议设置的. 在算法的对比实验中, 实验结果为 10 次 5 折交叉验证的平均值. 对比实验的 *NMI* 结果如表 2 所示, *NE* 结果如表 3 所示.

从表 2 对比实验可知, EBSKM 在大部分数据集中都表现出最佳效果. 可以证明所提出的模型在聚类任务中是有效的. 从表 3 可知由于 OMBC 算法是硬平衡聚类, 所以它的平衡性都达到了 1. 本文所出的算法相比传统算法在平衡性的表现上都是最好水平. 可以分析出加入平衡约束有助于算法模型更均匀划分. 数据集不平衡划分会导致在划分独立子空间时, 由于各簇

样本数量不同, 对于样本数量较少的簇会产生偏差从而影响聚类精度.

3.4 实验参数设置

本节主要讨论算法中超参数α与γ的选择情况.

表 2 算法准确率对比

数据集	EBSKM	K-means	EWKM	W-K-means	OMBC	FSBC
Iris	0.6934	0.6833	**0.7634**	0.6851	0.6947	0.7010
Wine	**0.9176**	0.7877	0.7592	0.7664	0.8753	0.9029
Seeds	**0.7502**	0.7171	0.6344	0.6904	0.7112	0.7464
Trace	0.5238	**0.5637**	0.5541	0.5554	0.5173	0.5433
Plane	**0.8836**	0.8687	0.8458	0.8580	0.8792	0.8568
SMS	**0.7910**	0.4663	0.7712	0.4173	0.3850	0.4796
Digit	**0.6937**	0.6608	0.3751	0.6924	0.6822	0.6690
Face (four)	**0.6822**	0.5730	0.4919	0.5329	0.5300	0.6350
Symbols	**0.7737**	0.7504	0.7521	0.7645	0.7718	0.7670
FacesUCR	**0.4943**	0.4287	0.2958	0.4377	0.4796	0.4578
Jaffe	**0.9424**	0.8672	0.8762	0.8717	0.9009	0.9293
ElectricDevices	**0.1750**	0.1610	0.1731	0.1609	0.1452	0.1337
HandOutlines	**0.2227**	0.1218	0.1133	0.1643	0.1786	0.1727
Mallat	0.8868	0.8440	0.7447	0.8344	**0.9068**	0.8804
StarLightCurves	**0.6125**	0.6068	0.5309	0.6073	0.6097	0.6075

表 3 算法平衡性对比

数据集	EBSWKM	K-means	EWKM	W-K-means	OMBC	FSBC
Iris	**1**	0.9497	0.9427	0.9430	1	0.9702
Wine	**1**	0.9514	0.9404	0.9460	1	0.9968
Seeds	**0.9997**	0.9787	0.9537	0.9517	1	0.9911
Trace	**0.9995**	0.9261	0.9233	0.9305	1	0.9451
Plane	**0.9578**	0.9207	0.8876	0.9103	1	0.9452
SMS	**0.9999**	0.9548	0.9438	0.9380	1	0.9577
Digit	0.9808	0.9597	0.8550	0.9621	1	**0.9932**
Face (four)	**0.9999**	0.8777	0.7423	0.8571	1	0.9453
Symbols	**0.9527**	0.8900	0.8420	0.9045	1	0.9321
FacesUCR	**0.9499**	0.8298	0.6222	0.8511	1	0.9094
Jaffe	**0.9874**	0.9407	0.9031	0.9276	1	0.9687
ElectricDevices	**0.9133**	0.8331	0.8252	0.8440	1	0.8907
HandOutlines	**0.9799**	0.5179	0.6904	0.6226	1	0.9055
Mallat	**0.9640**	0.9340	0.7997	0.9353	1	0.9567
StarLightCurves	0.9775	0.9712	0.9437	0.9704	1	**0.9866**

(1) 讨论α的取值

探究超参数α对算法性能的影响, 我们在 0 到 1 之间, 间距设置为 0.01, 测试α不同取值对算法性能的影响. 分别在 Wine、Seeds 和 Digit 数据集进行测试.

图 1 为α不同取值对 EBSKM 算法的影响情况. 横坐标为α的取值, 纵坐标为对应的 *NMI* 值. 从图中可见α取值过大或过小都会使算法性能产生波动, 可以看出平衡约束和特征选择同时起作用的时候算法才可以趋向更好的性能. 因此, 本文将超参数α统一取为 0.6.

(2) 讨论γ的取值

研究超参数γ对算法性能的影响, 我们在 1 到 500 之间, 间距设置为 10, 测试γ的不同取值对算法性能的影响.分别在 Wine、Seeds 和 Digit 数据集进行测试.

图 2 为γ不同取值对 EBSKM 算法的影响情况. 其横坐标为γ的取值, 纵坐标为对应的 *NMI* 值. 从图中可以看出γ取值过小时, 会使算法性能产生不稳定的结果. 因此, 本文将超参数γ统一取为 200.

4 结论与展望

针对 EWKM 算法, 聚类结果不平衡的问题, 本文提出将平衡约束加入到原本算法模型中, 提出在独立子空间上的软平衡聚类算法 EBSKM 算法, 通过 UCI 和 UCR 等数据集上进行实验, 所提出的算法在聚类性能和平衡性能上均展现出更好的效果. 但是, 实验过程中参数选择会直接影响算法性能, 因此如何合理选取算法中的参数, 是进一步要解决的问题.

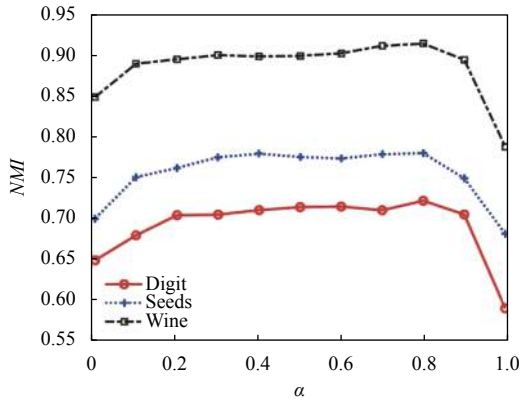

图 1 α对算法性能的影响

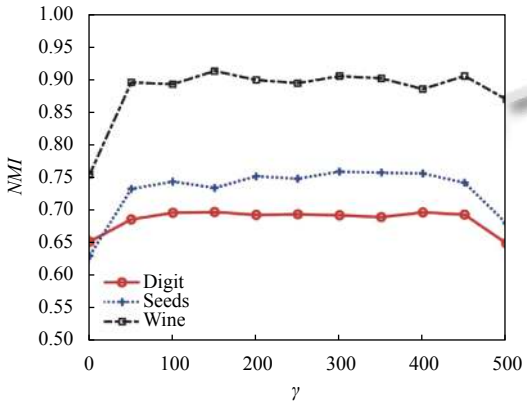

图 2 γ对算法性能的影响

参考文献

1 Hu JH, Pei J. Subspace multi-clustering: A review. Knowledge and Information Systems, 2018, 56(2): 257–284. [doi: 10.1007/s10115-017-1110-9]

2 Chen LF, Wang SR, Wang KJ, *et al*. Soft subspace clustering of categorical data with probabilistic distance. Pattern Recognition, 2016, 51: 322–332. [doi: 10.1016/j.patcog.2015.09.027]

3 Jing LP, Ng MK, Huang JZ. An entropy weighting K-means algorithm for subspace clustering of high-dimensional sparse data. IEEE Transactions on Knowledge and Data Engineering, 2007, 19(8): 1026–1041. [doi: 10.1109/TKDE.2007.1048]

4 Xiong LY, Wang C, Huang XH, *et al*. An entropy regularization K-means algorithm with a new measure of between-cluster distance in subspace clustering. Entropy, 2019, 21(7): 683. [doi: 10.3390/e21070683]

5 Yang MS, Nataliani Y. A feature-reduction fuzzy clustering algorithm based on feature-weighted entropy. IEEE Transactions on Fuzzy Systems, 2018, 26(2): 817–835. [doi: 10.1109/TFUZZ.2017.2692203]

6 Huang XH, Yang XF, Zhao JH, *et al*. A new weighting K-means type clustering framework with an l^2-norm regularization. Knowledge-Based Systems, 2018, 151: 165–179. [doi: 10.1016/j.knosys.2018.03.028]

7 Rajpoot P, Dwivedi P. Optimized and load balanced clustering for wireless sensor networks to increase the lifetime of WSN using MADM approaches. Wireless Networks, 2020, 26(1): 215–251. [doi: 10.1007/s11276-018-1812-2]

8 Patooghy A, Kamarei M, Farajzadeh A, *et al*. Load-balancing enhancement by a mobile data collector in wireless sensor networks. International Journal on Smart Sensing and Intelligent Systems, 2014, 7(5): 1–5.

9 Nallusamy R, Duraiswamy K, Dhanalaksmi R, *et al*. Optimization of non-linear multiple traveling salesman problem using K-means clustering, shrink wrap algorithm and meta-heuristics. International Journal of Nonlinear Science, 2010, 9(2): 171–177.

10 Banerjee A, Ghosh J. Frequency-sensitive competitive learning for scalable balanced clustering on high-dimensional hyperspheres. IEEE Transactions on Neural Networks, 2004, 15(3): 702–719. [doi: 10.1109/TNN.2004.824416]

11 Tang W, Yang Y, Zeng LL, *et al*. Size constrained clustering with MILP formulation. IEEE Access, 2020, 8: 1587–1599. [doi: 10.1109/ACCESS.2019.2962191]

12 Banerjee A, Ghosh J. On scaling up balanced clustering algorithms. Proceedings of the 2002 2nd SIAM International Conference on Data Mining. Arlington:SDM, 2002: 333–349.

13 王哲昀, 胡文军, 徐剑豪, 等. 标签分布熵正则的模糊 C 均值平衡聚类方法. 控制与决策, 2021: 1–7. [doi: 10.13195/j.kzyjc.2021.0398, 2021-07-02]

14 Costa LR, Aloise D, Mladenović N. Less is more: Basic variable neighborhood search heuristic for balanced minimum sum-of-squares clustering. Information Sciences, 2017, 415–416: 247–253.

15 Tang W, Yang Y, Zeng LL, *et al*. Optimizing MSE for clustering with balanced size constraints. Symmetry, 2019, 11(3): 338. [doi: 10.3390/sym11030338]

16 Zhou P, Chen JY, Fan MY, *et al*. Unsupervised feature selection for balanced clustering. Knowledge-Based Systems, 2020, 193: 105417. [doi: 10.1016/j.knosys.2019.105417]

17 Huang JZ, Ng MK, Rong HQ, *et al*. Automated variable weighting in K-means type clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2005, 27(5): 657–668. [doi: 10.1109/TPAMI.2005.95]

(校对责编: 牛欣悦)