完全自适应的谱聚类算法

谢娟英,丁丽娟
(陕西师范大学计算机科学学院,陕西西安 710062)

摘 要: 针对谱聚类算法 self-tuning 的局部尺度参数 σ_i 会受噪音点影响,进而影响聚类结果,及其所使用的 K-means 算法的不稳定,对聚类结果的影响,提出两种完全自适应的谱聚类算法 SC_SD(Spectral Clustering based on Standard Deviation)和 SC_MD(Spectral Clustering based on Mean Distance),分别定义样本 i 的标准差、样本 i 到其余样本的距离均值,为样本 i 的邻域半径,统计邻域内的样本数,以样本 i 的邻域标准差为其局部尺度参数,避免样本 i 的局部尺度参数受噪音点影响,进而影响聚类结果;以方差优化初始聚类中心的 SD_K-medoids 算法代替 K-means 算法,克服 K-means 算法的不稳定,发现数据的真实分布. UCI 数据集和人工数据集实验测试表明,提出的 SC_SD 和 SC_MD 算法能得到更优聚类结果,不受噪音点影响,有很好的伸缩性. 提出的 SC_SD 和 SC_MD 能完全自适应地发现数据集的真实分布信息,尤其 SC_MD 算法很适合较大规模数据集的聚类分析.

关键词: 谱聚类; 邻域; 标准差; 均值; 自适应
中图分类号: TP181 **文献标识码:** A **文章编号:** 0372-2112(2019)05-1000-09
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2019.05.004

The True Self-adaptive Spectral Clustering Algorithms

XIE Juan-ying, DING Li-juan
(*School of Computer Science, Shaanxi Normal University, Xi'an, Shaanxi* 710062, *China*)

Abstract: To avoid the clustering results with the local scaling parameter σ_i of self-tuning may be influenced by outliers, and the unstable clustering results from K-means in self-tuning, two true self-adaptive spectral clustering algorithms were proposed. The two spectral clustering algorithms are respectively named as SC_SD(Spectral Clustering based on Standard Deviation) and SC_MD(Spectral Clustering based on Mean Distance). They respectively define the standard deviation of point i, and the mean distance from point i to others, as its radius of neighborhood, then count the number of points in the neighborhood, and use the standard deviation of point i in the neighborhood as its local scaling parameter, so as to avoid the influence from outliers to the local scaling parameter σ_i of point i, and the distortion in clustering results of self-tuning. SD_K-medoids are adopted to instead of K-means in self-tuning to avoid the unstable clustering results of K-means, so as to get the true clustering of a dataset. The experimental results on UCI datasets and on synthetic datasets demonstrate that SC_SD and SC_MD can obtain better clustering results than that of traditional spectral clustering algorithm NJW and spectral clustering algorithm self-tuning, and are robust to noises, and has got good scalability. The proposed SC_SD and SC_MD can detect the clustering of a dataset without any given information, and the SC_MD can be used to detect the clustering of a comparable big data.

Key words: spectral clustering; neighborhood; standard deviation; mean distance; self-adaption

1 引言

聚类分析作为数据挖掘的一项重要技术是发现和探索事物内在联系的有效手段,在各领域得到广泛应用[1]. 聚类根据一定的相似性原则将数据对象划分为若干子集,每个子集构成一个类簇,簇内对象彼此相似,但与其他簇的对象相异[2,3]. 聚类作为无监督学习方法,可在没有任何先验知识的情况下,发现无标记数据

收稿日期:2018-02-05;修回日期:2018-07-22;责任编辑:蓝红杰
基金项目:国家自然科学基金(No. 61673251);国家重点研发计划(No. 2016YFC0901900);中央高校基本科研业务费专项资金(No. GK2017010006);研究生培养创新基金(No. 2015CX028, No. 2016CSY009)

的内部结构[4]. K-means[5]、FCM[6]、PAM[7]等构成经典的基于划分的聚类算法,但这些算法适合发现球状簇,无法发现非凸形状的簇,且易于陷入局部最优而不能发现数据集的真实分布.

谱聚类(Spectral Clustering,SC)是一种基于谱图理论的聚类算法[8],能发现任意形状的簇且收敛于全局最优解. 谱聚类算法以样本为顶点,样本相似性为顶点间连接边的权重,得到表达数据集样本及其相似性的无向带权图,将聚类问题转化为图划分问题. 经典谱聚类算法有 PF 算法[9]、SM 算法[10]和 NJW 算法[11]等. NJW 算法在构建样本相似性矩阵时需要人为给定尺度参数 σ,σ 的不同取值得到不同聚类结果. 如何选择合适的尺度参数 σ 引起了诸多学者关注[12~15]. 著名的 self-tuning 算法[12]针对 NJW 算法的尺度参数选择问题,提出针对样本 i 的自适应尺度参数 σ_i,代替 NJW 算法构建相似性矩阵的全局尺度参数 σ,使得相似性矩阵尽可能反映数据集样本的真实分布. 但 self-tuning 算法将样本 i 的尺度参数 σ_i 定义为样本 i 到第 p 个近邻的距离,致使样本 i 的尺度参数 σ_i 可能会受到离群点影响,且参数 p 的选择没有理论依据,只依赖于经验值. 针对 self-tuning 算法的局限,Xie 等人[15]提出了基于样本局部标准差的谱聚类算法,以避免 self-tuning 算法的局限,大量实验证实了所提算法的优越性,但该算法没有解决参数 p 依赖于经验值的问题.

为此,本文提出基于样本邻域标准差的完全自适应的谱聚类算法,根据数据集样本空间分布,自适应地确定样本 i 的尺度参数,使谱聚类算法的相似性矩阵能尽可能准确地反映数据集样本的实际分布信息,以发现数据集样本的潜在分布规律,消除人为给定参数的主观性,使聚类结果符合数据实际分布;算法还引入 SD_K-medoids 算法[16]代替谱聚类算法中的 K-means 算法,以解决 K-means 引起的谱聚类算法的聚类结果不稳定问题. 真实数据集和人工数据集实验测试表明,提出的基于样本邻域标准差的完全自适应的谱聚类算法,解决了谱聚类算法的参数选择难题,同时解决了谱聚类算法的聚类结果不稳定问题,能更有效地发现数据集的真实类簇分布.

2　self-tuning 算法及其改进

谱聚类算法将聚类问题转化为一个无向图 $G=(\boldsymbol{V},\boldsymbol{E})$的多路划分问题,其中,$\boldsymbol{V}=\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n\}$是所有 n 个样本集合,构成无向图顶点集;$E=\{A_{ij}\}$是无向图顶点间的边权集合. 待聚类样本间的相似性矩阵 $\boldsymbol{A}=(A_{ij}),i,j=1,\cdots,n$,又称亲和矩阵,是无向图的邻接矩阵,$A_{ij}=\begin{cases}\exp(-d^2(\boldsymbol{x}_i,\boldsymbol{x}_j)/\sigma^2), & i\neq j\\ 0, & i=j\end{cases}$,包含了聚类需要的所有信息. self-tuning 算法[12]针对传统谱聚类算法计算亲和矩阵的全局尺度参数 σ 不能准确反映数据集样本真实分布信息的问题,提出自适应的局部尺度参数 σ_i,根据样本 i 的第 p 个近邻定义样本 i 的局部尺度参数 σ_i,使 $\sigma_i=d(\boldsymbol{x}_i,\boldsymbol{x}_p)$,$d(\boldsymbol{x}_i,\boldsymbol{x}_p)$为样本 i,p 间的欧氏距离,反映样本相似性的亲和矩阵定义为式(1). 实验证明[12] self-tuning 算法的性能优于 NJW 算法.

$$A_{ij}=\begin{cases}\exp(-d^2(\boldsymbol{x}_i,\boldsymbol{x}_j)/\sigma_i\sigma_j), & i\neq j\\ 0, & i=j\end{cases} \tag{1}$$

然而,self-tuning 算法的局部尺度参数 σ_i 可能会受到离群点影响. 针对此问题,基于局部标准差的谱聚类算法[15]定义样本 i 的局部尺度参数 σ_i 为式(2),相应的亲和矩阵定义为式(3),其中的 σ_{std_i}为样本 i 的前 p 个近邻的局部标准差,$d(\boldsymbol{x}_i,\boldsymbol{x}_j)$为样本 i,j 间的欧氏距离. 大量实验测试证实[15],基于局部标准差的谱聚类算法,聚类效果与时间均优于 self-tuning 算法,但其参数 p 依然需要人为给定.

$$\sigma_{std\_i}=\sqrt{\frac{1}{p-1}\sum_{t=1}^{p}d^2(\boldsymbol{x}_i,\boldsymbol{x}_t)} \tag{2}$$

$$A_{ij}=\begin{cases}\exp(-d^2(\boldsymbol{x}_i,\boldsymbol{x}_j)/\sigma_{std\_i}\sigma_{std\_j}), & i\neq j\\ 0, & i=j\end{cases} \tag{3}$$

3　本文自适应谱聚类算法

设数据集为 $\boldsymbol{X}=\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n\}$,样本 $\boldsymbol{x}_i$ 有 m 维特征,即 $\boldsymbol{x}_i=(x_{i1},\cdots,x_{im})$,$(i=1,2,\cdots,n)$. 为消除样本不同属性的不同量纲对实验结果的影响,采用式(4)最大最小标准化方法对样本进行标准化,使样本属性落入[0,1]. 式(4)的 $\boldsymbol{x}_{:,j}$表示所有样本在第 j 个属性的取值.

$$x'_{i,j}=\frac{x_{i,j}-\min(\boldsymbol{x}_{:,j})}{\max(\boldsymbol{x}_{:,j})-\min(\boldsymbol{x}_{:,j})} \tag{4}$$

3.1　算法思想与描述

本文谱聚类算法根据样本分布信息,完全自适应地定义样本 i 的局部尺度参数 σ_i,以使样本的亲和矩阵尽可能准确地反映样本分布信息. 为此,定义式(5)样本 i 的标准差 std_i,式(6)样本 i 到其他样本的距离均值 d_{mean_i},式中 $d(\boldsymbol{x}_j,\boldsymbol{x}_i)$为样本 j,i 间的欧氏距离,n 为总样本数. 分别以 std_i 和 d_{mean_i}为样本 i 的邻域半径,统计样本 i 邻域内的样本数,以样本 i 在相应邻域的局部标准差作为样本 i 的局部尺度参数 σ_i,得到样本 i 的完全自适应的局部尺度参数,避免 self-tuning 算法的局部尺度参数可能受噪音点干扰的缺陷,同时避免基于局部标准差的谱聚类算法的参数 p 依赖于经验值的缺陷.

$$std_i=\sqrt{\frac{1}{n-1}\sum_{j=1}^{n}d^2(\boldsymbol{x}_j,\boldsymbol{x}_i)} \tag{5}$$

$$d_{mean\_i}=\frac{1}{n-1}\sum_{j=1,j\neq i}^{n}d(\boldsymbol{x}_j,\boldsymbol{x}_i) \tag{6}$$

令 r_{SD_i} 和 r_{MD_i} 分别表示样本 i 的标准差邻域半径和均值邻域半径,即 $r_{SD\_i}=std_i$, $r_{MD\_i}=d_{mean\_i}$. S_i 和 M_i 分别为样本 i 对应邻域半径分别为 r_{SD_i} 和 r_{MD_i} 的邻域内样本数,则分别定义样本 i 的完全自适应的局部尺度参数 σ_{SD_i} 和 σ_{MD_i} 为式(7)和(8),亲和矩阵为式(9)和(10).从而得到两种完全自适应的谱聚类算法,分别命名为 SC_SD(Spectral Clustering based on Standard Deviation)和 SC_MD(Spectral Clustering based on Mean Distance).该两个谱聚类算法的详细步骤描述如下.

$$\sigma_{SD\_i}=\sqrt{\frac{1}{S_i-1}\sum_{j=1}^{S_i}d^2(\boldsymbol{x}_j,\boldsymbol{x}_i)} \tag{7}$$

$$\sigma_{MD\_i}=\sqrt{\frac{1}{M_i-1}\sum_{j=1}^{M_i}d^2(\boldsymbol{x}_j,\boldsymbol{x}_i)} \tag{8}$$

$$A_{ij}=\begin{cases}\exp(-d^2(\boldsymbol{x}_i,\boldsymbol{x}_j)/\sigma_{SD\_i}\sigma_{SD\_j}), & i\neq j\\ 0, & i=j\end{cases} \tag{9}$$

$$A_{ij}=\begin{cases}\exp(-d^2(\boldsymbol{x}_i,\boldsymbol{x}_j)/\sigma_{MD\_i}\sigma_{MD\_j}), & i\neq j\\ 0, & i=j\end{cases} \tag{10}$$

完全自适应的谱聚类算法详细步骤描述如算法 1 所示.

算法 1

输入:$dataset=\boldsymbol{X},\boldsymbol{X}=\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n\}\in\boldsymbol{R}^{n\times m}$

输出:K 个类簇

Step1:分别根据式(9)、(10)构建亲和矩阵 $\boldsymbol{A}\in\boldsymbol{R}^{n\times n}$;

Step2:计算亲和矩阵的度矩阵 $\boldsymbol{D}$ 和拉普拉斯矩阵 $\boldsymbol{L}=\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$;

Step3:令 $\lambda_1\geq\lambda_2\geq\cdots\geq\lambda_K\geq0$ 是 $\boldsymbol{L}$ 的前 K 个最大特征值,$\boldsymbol{v}^1,\boldsymbol{v}^2,\cdots,\boldsymbol{v}^K$ 是前 K 大特征值对应的特征向量,则构造矩阵 $\boldsymbol{V}=[\boldsymbol{v}^1,\boldsymbol{v}^2\cdots,\boldsymbol{v}^K]\in\boldsymbol{R}^{n\times K}$,其中 $\boldsymbol{v}^i$ 是列向量,$i=1,2,\cdots,K$;

Step4:按行对矩阵 $\boldsymbol{V}$ 进行归一化,得到矩阵 $\boldsymbol{U}$,即

$$u_{ij}=v_{ij}\Big/\left(\sum_k v_{ik}^2\right)^{\frac{1}{2}};$$

Step5:将 $\boldsymbol{U}$ 中的每一行当作一个样本,采用文献[16]的 SD_K-medoids 算法对样本进行聚类;

Step6:将原始样本 $\boldsymbol{x}_i$ 分配到第 j 个类簇,当且仅当矩阵 $\boldsymbol{U}$ 的第 i 行属于第 j 个类簇.

3.2 算法时间复杂度分析

3.1 节算法的六个步骤中,彼此无嵌套调用关系,其时间复杂度取决于时间复杂度最高的一步.第一步建立亲和矩阵的时间复杂度是 $O(n^2)$,n 为样本数;第二步中计算拉普拉斯矩阵的时间复杂度为 $O(n^3)$;第三步的时间复杂度为 $O(K^2)$,K 为类簇数;第四步时间复杂度为 $O(Kn)$;第五步 SD_K-medoids 算法的时间复杂度为 $O(n^2+Knt)$,t 为迭代次数;第六步的时间复杂度为 $O(n)$.实际实验中,将 $\boldsymbol{L}=\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$ 改写成式(11)形式,优化计算拉普拉斯矩阵的过程,使第二步的时间复杂度降为 $O(n^2)$.因此,本文谱聚类算法的时间复杂度为 $O(n^2)$.

$$\boldsymbol{L}=\begin{bmatrix}d_1&&&\\&d_2&&\\&&\ddots&\\&&&d_n\end{bmatrix}^{-\frac{1}{2}}\begin{bmatrix}a_{11}&a_{12}&\cdots&a_{1n}\\a_{21}&a_{22}&\cdots&a_{2n}\\\vdots&\vdots&\ddots&\vdots\\a_{n1}&a_{n2}&\cdots&a_{nn}\end{bmatrix}\cdot\begin{bmatrix}d_1&&&\\&d_2&&\\&&\ddots&\\&&&d_n\end{bmatrix}^{-\frac{1}{2}}=(d_i^{-\frac{1}{2}}a_{ij}d_i^{-\frac{1}{2}})_{i,j=1,\cdots,n} \tag{11}$$

4 实验结果与分析

实验代码使用 MATLAB R2017b 实现;实验主要环境为 Win10 64bit 操作系统,8GB 内存,Intel(R) Core(TM) i5-6600 CPU @3.30GHz 3.31GHz;伸缩性实验与人工数据集 Dataset4 的实验在集群 Linux version 3.10.0-327,Architecture:x86_64,Model name:Intel(R) Xeon(R) CPU E5-2609 v3 @ 1.90GHz,CPU MHz:1200.042,CPU(s):12 实现.

实验首先测试引入 SD_K-medoids 算法替换谱聚类算法中的 K-means 算法的有效性.为此,用 SD_K-medoids 算法替换 NJW、self-tuning 算法步骤 5 中的 K-means,得到 NJW+和 self-tuning+算法,比较 NJW、self-tuning、NJW+和 self-tuning+算法的性能.然后测试提出的完全自适应的谱聚类算法 SC_SD 和 SC_MD 的性能.

实验数据包括 UCI 机器学习数据库[17]测试聚类算法常用的数据集,以及测试聚类算法的经典人工数据集和我们特意生成的具有挑战性的人工数据集.各算法的聚类性能分别采用聚类准确率 Acc(clustering Accuracy)、调整互信息 AMI(Adjusted Mutual Information)和调整 Rand 指数 ARI(Adjusted Rand Index)三种经典的聚类算法评价指标进行评价.三种指标的上界均为 1,值越大表示聚类结果越好[18,19].

NJW 算法需要给定全局尺度参数 σ,self-tuning 需要给定近邻样本数 p.参数的不确定性以及 K-means 的不稳定性均会影响实验结果,为此,经过多次实验,选择各算法聚类准确率最优时的参数,比较各算法在最优参数下重复运行 20 次的平均 Acc、AMI 和 ARI.实验中数据集类簇数 K 为其真实类簇数,NJW、self-tuning 和 NJW+算法的尺度参数是其最优的尺度参数,self-tuning+的参数和 self-tuning 相同.

4.1 UCI 数据集实验

实验使用的 UCI 真实数据集描述见表 1.其中 Waveform40 是 Waveform21 增加 19 个噪音属性形成的数据集.表 2 是 NJW、self-tuning、NJW+和 self-tuning+

算法的聚类结果比较. 表 3 是 NJW、self-tuning 和提出的完全自适应的谱聚类算法 SC_SD 和 SC_MD 的聚类结果比较. 表 4 为 NJW、self-tuning、SC_MD 和 SC_SD 在各数据集运行 20 次的平均时间. 加粗表示最优结果.

表 1　实验使用的 UCI 数据集描述

数据集	样本数	属性个数	类簇数
Iris	150	4	3
Mice	1080	81	2
Pima-indians-diabetes	768	8	2
Soybean-small	47	35	4
Ionosphere	351	34	2
Segmentation-test	2310	19	7
Waveform21	5000	21	3
Waveform40	5000	40	3

表 2　SD_K-medoids 算法对 NJW 和 self-tuning 聚类结果的影响

Iris				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=0.5$	0.8510	0.7172	0.6892
self-tuning	$p=10$	0.8807	0.7532	0.7442
NJW +	$\sigma=0.35$	0.8867	0.7331	0.7163
self-tuning +	$p=10$	**0.9000**	**0.7667**	**0.7445**
Mice				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=1$	0.5318	0.0023	0.0029
self-tuning	$p=5$	**0.5556**	**0.0301**	**0.0095**
NJW +	$\sigma=1$	0.5306	0.0022	0.0007
self-tuning +	$p=5$	**0.5556**	**0.0301**	**0.0095**
Pima-indians-diabetes				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=0.35$	**0.6810**	0.0676	0.1233
self-tuning	$p=20$	0.6784	0.0731	0.1229
NJW +	$\sigma=0.35$	0.6523	0.0008	0.0023
self-tuning +	$p=20$	**0.6810**	**0.0763**	**0.1268**
Soybean-small				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=1$	0.8851	0.8145	0.7823
self-tuning	$p=15$	0.9032	0.8932	0.8683
NJW +	$\sigma=1$	0.8298	0.7322	0.6738
self-tuning +	$p=15$	**1**	**1**	**1**
Ionosphere				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=0.15$	0.7051	0.0932	0.1186
self-tuning	$p=10$	**0.7265**	0.2289	**0.2016**
NJW +	$\sigma=0.15$	0.7123	0.1055	0.1300
self-tuning +	$p=10$	0.7208	0.2502	0.1902
Segmentation-test				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=1.5$	0.6160	0.6023	0.4753
self-tuning	$p=10$	0.5338	0.5831	0.4293
NJW +	$\sigma=1.5$	**0.6628**	0.5748	0.4582
self-tuning +	$p=10$	0.6545	**0.6686**	**0.5458**

续表

Waveform21				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=1.5$	0.5060	0.3692	0.2500
self-tuning	$p=15$	0.5042	0.3629	0.2499
NJW +	$\sigma=1.5$	**0.5068**	**0.3693**	**0.2502**
self-tuning +	$p=20$	0.5052	0.3620	0.2495
Waveform40				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=0.5$	**0.5164**	0.3666	**0.2509**
self-tuning	$p=25$	0.5138	**0.3670**	0.2499
NJW +	$\sigma=0.5$	0.5150	0.3665	0.2507
self-tuning +	$p=25$	0.5134	0.3669	0.2498

表 2 实验结果揭示,SD_K-medoids 聚类算法对 NJW 和 self-tuning 算法都有性能上的提升,除了包含 19 个噪音属性的 Waveform40 数据集. 这说明:①采用 SD_K-medoids 算法代替 NJW 和 self-tuning 算法中的 K-means 不仅是可行的,而且是有效的;②K-means 算法的随机性有可能使谱聚类算法避免噪音属性对聚类结果的影响.

表 3 实验结果揭示,除了在 Waveform21 数据集外,提出的完全自适应的谱聚类算法 SC_SD 和 SC_MD 在不需要人为设定任何参数的情况下,在各数据集的聚类性能绝对地优于经典谱聚类算法 NJW 和自适应谱聚类算法 self-tuning 在其最优参数时的结果. 在 Waveform21 数据集,NJW 算法的性能在各个指标都最优,本文 SC_SD 算法在该数据集的聚类准确率 Acc 和调整互信息 AMI 优于 self-tuning 算法,位居第二,self-tuning 算法的 ARI 指标优于 SC_SD,位居第二,提出的 SC_MD 在该数据集的 Acc 和 ARI 指标最差,但在 AMI 指标优于 self-tuning,位居第三. 尽管 NJW 在 Waveform21 数据集的各指标都取得最优,但取得该结果的最佳尺度参数很难寻找,实验中经过多次试探得到.

以上 UCI 机器学习数据库数据集的聚类结果指标比较揭示,提出的完全自适应的谱聚类算法 SC_SD 和 SC_MD 有很好的聚类性能.

表 4 关于 NJW、self-tuning、SC_MD 与 SC_SD 算法在 UCI 数据集的平均运行时间比较显示:NJW 算法运行时间最短,SC_SD 算法运行时间最长,SC_MD 与 self-tuning 算法的运行时间居中,且相差不大. 分析原因是:NJW 算法的尺度参数人为给定,而 SC_SD、SC_MD 与 self-tuning 算法的局部尺度参数需要计算得到. self-tuning 算法为计算样本 i 的局部应尺度参数 σ_i,需计算样本 i 与第 p 个近邻的距离,而为了找到第 p 个近邻,需要计算样本 i 与所有样本的距离,并对距离进行排序. SC_MD 算法需计算样本 i 与所有样本点间距离均值,作为样本 i 的邻域半径,然后计算样本 i 在邻域半径内的标准差作为其局部尺度参数 σ_{MD_i}. SC_MD 比 self-tuning 的计算量少了排序样本 i 与所有样本的距离,但多了统计样本 i 邻域内的样本数,因此,俩算法

计算量差别不大. SC_SD 算法首先计算样本 i 的标准差作为邻域半径,然后统计邻域内的样本数,计算样本 i 的邻域标准差作为其局部尺度参数 σ_{SD_i}. SC_SD 算法比 SC_MD 算法花费更多时间计算邻域半径.

表 3 NJW,self-tuning,SC_SD 与 SC_MD 在 UCI 数据集的聚类结果

Iris				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=0.5$	0.8510	0.7172	0.6892
self-tuning	$p=10$	0.8807	**0.7532**	**0.7442**
SC_SD	–	**0.8867**	0.7331	0.7163
SC_MD	–	**0.8867**	0.7331	0.7163
Mice				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=1$	0.5318	0.0023	0.0029
self-tuning	$p=5$	0.5556	**0.0301**	0.0095
SC_SD	–	0.5796	0.0163	0.0244
SC_MD	–	**0.5815**	0.0171	**0.0256**
Pima-indians-diabetes				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=0.35$	0.6810	0.0676	0.1233
self-tuning	$p=20$	0.6784	0.0731	0.1229
SC_SD	–	**0.6849**	0.0785	**0.1320**
SC_MD	–	0.6810	**0.0800**	0.1277
Soybean-small				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=1$	0.8851	0.8145	0.7823
Self-tuning	$p=15$	0.9032	0.8932	0.8683
SC_SD	–	**1**	**1**	**1**
SC_MD	–	**1**	**1**	**1**
Ionosphere				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=0.15$	0.7051	0.0932	0.1186
self-tuning	$p=10$	0.7265	**0.2289**	0.2016
SC_SD	–	**0.7322**	0.1892	**0.2135**
SC_MD	–	0.7265	0.1870	0.2029
Segmentation-test				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=1.5$	0.6160	0.6023	0.4753
self-tuning	$p=10$	0.5338	0.5831	0.4293
SC_SD	–	**0.7632**	**0.6704**	**0.5895**
SC_MD	–	0.7048	0.6299	0.5158
Waveform21				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=1.5$	**0.5060**	**0.3692**	**0.2500**
self-tuning	$p=15$	0.5042	0.3629	0.2499
SC_SD	–	0.5044	0.3668	0.2497
SC_MD	–	0.5040	0.3658	0.2494
Waveform40				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=0.5$	0.5164	0.3666	0.2509
self-tuning	$p=25$	0.5138	0.3670	0.2499
SC_SD	–	**0.5170**	**0.3682**	0.2503
SC_MD	–	0.5168	**0.3682**	0.2503

表 4 NJW,self-tuning,SC_SD 与 SC_MD 在 UCI 数据集的运行时/s

数据集	NJW	self-tuning	SC_MD	SC_SD
Iris	**0.0145**	0.0228	0.0291	0.0517
mice	**0.7848**	1.3291	1.5965	4.0373
Pima-indians-diabetes	**0.0982**	0.2523	0.5177	1.6877
Soybean-small	0.0063	0.0121	**0.0049**	0.0067
ionosphere	**0.0542**	0.0851	0.1378	0.2828
Segmentation-test	**8.6529**	13.0308	11.7201	42.7923
Waveform21	**14.6588**	63.3561	31.0623	238.4142
Waveform40	**14.6964**	63.5291	31.2982	233.8996

4.2 人工数据集实验

本节采用测试聚类算法性能常用的经典人工数据集 R15[20]、Path-based1[21] 和 S1[22],以及随机生成的 4 个带有刁难性的人工数据集对提出的 SC_SD 和 SC_MD 算法的聚类性能进行测试. 各数据集详细信息见表 5. 人工模拟数据集 Dataset1、Dataset2、Dataset3 和 Dataset4 的生成参数分别见表 6、表 7、表 8 和表 9.

将提出的 SC_SD 和 SC_MD 算法分别在表 5 的 7 个人工数据集运行,将聚类结果与 NJW 算法、self-tuning 算法在其最优参数下重复运行 20 次的平均聚类结果进行比较. 表 10 展示了各算法在 7 个人工数据集的聚类结果指标值比较. 表 11 给出了各算法的运行时间比较. 加粗表示最优结果.

表 5 实验用人工模拟数据集

数据集	样本数	属性个数	类簇数
R15	600	2	15
Path-based1	300	2	3
S1	5000	2	15
Dataset1	678	2	3
Dataset2	3000	2	2
Dataset3	12000	2	3
Dataset4	20000	2	2

表 6 Dataset1 参数表

参数	Cluster1	Cluster2	Cluster3
mean	[3,2]	[8,2]	[6,5.5]
covariance	$\begin{bmatrix}0.7 & 0\\ 0 & 0.7\end{bmatrix}$	$\begin{bmatrix}0.7 & 0\\ 0 & 0.7\end{bmatrix}$	$\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$
# points	300	300	78

表 7 Dataset2 参数表

参数	Cluster1	Cluster2
	$\theta\in[0,2\pi]$	
Attribute1	$x_1\in[2\sin\theta,3\sin\theta]$	$x_2\in[10\sin\theta,11\sin\theta]$
Attribute2	$y_1\in[2\cos\theta,3\cos\theta]$	$y_2\in[10\cos\theta,11\cos\theta]$
# points	1500	1500

表 8　Dataset3 参数表

参数	Cluster1	Cluster2	Cluster3
Attribute1	$x_1 \in [0,100]$	$x_2 \in [50,60]$	$x_3 \in [20,80]$
Attribute2	$y_1 \in [0,10]$	$y_2 \in [10,50]$	$y_3 \in [50,60]$
# points	4000	4000	4000

表 9　Dataset4 构造参数表

参数	Cluster1	Cluster2
Attribute1	$x_1 \in \left[-\frac{3}{2},\frac{3}{2}\right]$	$x_2 \in \left[-\frac{0.75}{4.4},\frac{0.25}{4.4}\right]$
Attribute2	$y_1 \in \left[\frac{1}{2}x_1^{\ 2}-\frac{1}{6},\frac{1}{2}x_1^{\ 2}+\frac{1}{6}\right]$	$y_2 \in [0.95,1.95]$
# points	12000	8000

表 10　各算法在表 5 的人工模拟数据集的聚类结果比较

R15				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=0.05$	0.7962	0.8897	0.7858
self-tuning	$p=5$	0.6891	0.8033	0.5642
SC_SD	–	**0.9917**	**0.9845**	**0.9820**
SC_MD	–	0.9167	0.9475	0.9054
Path-based1				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=1$	0.7600	0.5294	0.4797
self-tuning	$p=20$	**0.7700**	**0.5417**	**0.4921**
SC_SD	–	0.7033	0.4450	0.3965
SC_MD	–	0.7667	0.5380	0.4882
S1				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=0.35$	0.8663	0.9235	0.8569
self-tuning	$p=10$	**0.9715**	**0.9746**	**0.9605**
SC_SD	–	0.7510	0.8795	0.7698
SC_MD	–	0.7248	0.8429	0.7170
Dataset1				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=0.5$	**0.9868**	**0.9302**	**0.9711**
self-tuning	$p=20$	0.9479	0.8868	0.9055
SC_SD	–	0.9706	0.8900	0.9350
SC_MD	–	0.9794	0.9126	0.9544
Dataset2				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=0.05$	**1**	**1**	**1**
self-tuning	$p=10$	**1**	**1**	**1**
SC_SD	–	**1**	**1**	**1**
SC_MD	–	**1**	**1**	**1**
Dataset3				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=0.5$	0.6401	0.5104	0.3880
self-tuning	$p=5$	0.6934	0.5317	0.4220
SC_SD	–	**0.7532**	**0.5614**	**0.5002**
SC_MD	–	0.7424	0.5531	0.4900
Dataset4				
算法	参数	Acc	AMI	ARI
NJW	$\sigma=0.35$	0.9999	0.9992	0.9998
self-tuning	$p=10$	**1**	**1**	**1**
SC_SD	–	**1**	**1**	**1**
SC_MD	–	**1**	**1**	**1**

从表 10 实验结果可见,提出的 SC_SD 和 SC_MD 算法在经典人工模拟数据集 R15 上的性能远远优于传统谱聚类算法 NJW 和自适应谱聚类算法 self-tuning. 提出的 SC_MD 算法在 Path-based1 上的性能略低于 self-tuning 算法在最佳尺度参数时的结果,但是优于 NJW 在最佳尺度参数时的结果,SC_SD 在该数据集的性能不如 SC_MD 和 NJW. 在 S1 数据集,提出的 SC_SD 和 SC_MD 算法的性能不如传统 NJW 和自适应的 self-tuning 谱聚类算法在最佳尺度参数时的聚类结果,self-tuning 算法在 S1 数据集表现出很好的聚类能力,说明 self-tuning 算法在刻意选择的最佳尺度参数下,可以取得很好的聚类效果,但其最佳尺度参数很难发现. 在 Dataset1 数据集,NJW 算法在最佳尺度参数时的聚类性能最好,提出的 SC_MD 和 SC_SD 算法分别位居第 2 和第 3,但不需要任何辅助信息,self-tuning 算法在其最佳尺度参数时的聚类结果不如其他三个谱聚类算法. 在 Dataset2 数据集,4 个谱聚类算法的聚类性能均达到最佳,但是提出的 SC_SD 和 SC_MD 算法不需要任何先验知识. 在 Dataset3 数据集,提出 SC_SD 和 SC_MD 算法的性能绝对地优于传统谱聚类算法 NJW 和自适应谱聚类算法 self-tuning 在其最佳尺度参数时的聚类结果. 在 Dataset4 数据集,提出的 SC_SD 和 SC_MD 算法,以及 self-tuning 算法均达到了最佳聚类结果,但提出的 SC_SD 和 SC_MD 算法不需要任何辅助信息,而 self-tuning 算法需要经过多次试探,选择其最佳尺度参数.

以上人工模拟数据集的聚类结果分析揭示,提出的 SC_SD 和 SC_MD 算法均能有效发现数据集的样本分布,尽管该两个算法在 S1 数据集的聚类结果不如 NJW 和 self-tuning 算法分别在其最优尺度参数时的聚类结果,但是,SC_SD 和 SC_MD 算法在不需要任何先验知识的情况下,所取得的聚类结果也是非常不错的.

表 11 关于各算法平均运行时间的比较显示,NJW 算法运行时间最少,本文提出 SC_SD 算法运行时间最长,self-tuning 算法在规模较小的 R15、Path-based1 和 Dataset1 数据集的运行时间少于提出的 SC_MD 算法,但是当数据集规模上千时,提出的 SC_MD 算法的收敛速度明显比 self-tuning 算法的收敛速度快.

表 11　各算法在表 5 的人工数据集运行时间比较/s

数据集	NJW	self-tuning	SC_MD	SC_SD
R15	**0.0558**	0.1313	0.3111	0.9481
Path-based1	**0.0163**	0.0372	0.0782	0.1796
S1	**16.7654**	72.8985	36.0284	289.0124
Dataset1	**0.0757**	0.2027	0.3954	1.1348
Dataset2	**3.8750**	16.3372	10.8242	73.3479
Dataset3	**458.7754**	1079.4568	725.3401	3551.4015
Dataset4	**708.9174**	7419.2738	1539.2134	19541.2680

由此可见,在同时考虑收敛速度和聚类效果的情

况下,SC_MD 是不错的谱聚类算法.

4.3 鲁棒性测试实验

本节将测试提出的 SC_SD 和 SC_MD 算法的鲁棒性,为此,随机生成规模为 1000 * 5 的有 5 个簇的二维人工数据集,数据集的生成参数如表 12 所示. 为了测试提出的 SC_SD 和 SC_MD 算法的鲁棒性,对数据集的中间簇 C3 加入服从二维正态分布,均值为[6,6],标准差为[1,1]的噪音,噪音比例依次为 5%、10%、15%、20%、25%、30%、35%、40%、45%,得到 9 个含不同比例噪音的数据集. 选择中间簇加入噪音的目的是,使带有噪音的簇与周围 4 个簇产生更多交叠,增加聚类难度,以便更好测试算法对噪音的鲁棒性能.

在包括原始数据集的共 10 个规模为 1000 * 5 的人工模拟数据集,分别运行提出的完全自适应的谱聚类算法 SC_SD 和 SC_MD,以及传统谱聚类算法 NJW 和自适应谱聚类算法 self-tuning,比较各算法的聚类准确率,聚类结果指标值 AMI 和 ARI,以及聚类时间. 通过试探实验,得到 NJW 的最佳尺度参数 $\sigma = 0.35$,self-tuning 的最佳参数 $p = 20$. 分别在最佳尺度参数下运行 NJW 和 self-tuning 各 20 次,将其平均性能与提出的 SC_SD 和 SC_MD 算法进行比较. 图 1 是各算法的聚类结果指标值和运行时间比较. 表 13 是 SC_SD 和 SC_MD 算法与 NJW + 和 self-tuning + 算法对原始不含噪音数据集的实验结果,加粗表示最优结果.

表 12 鲁棒性实验人工数据集生成参数

类簇	均值	标准差	噪音标准差
C1	[3,9]	[1,1]	–
C2	[9,9]	[1,1]	–
C3	[6,6]	[0.6,0.6]	[1,1]
C4	[3,3]	[1,1]	–
C5	[9,3]	[1,1]	–

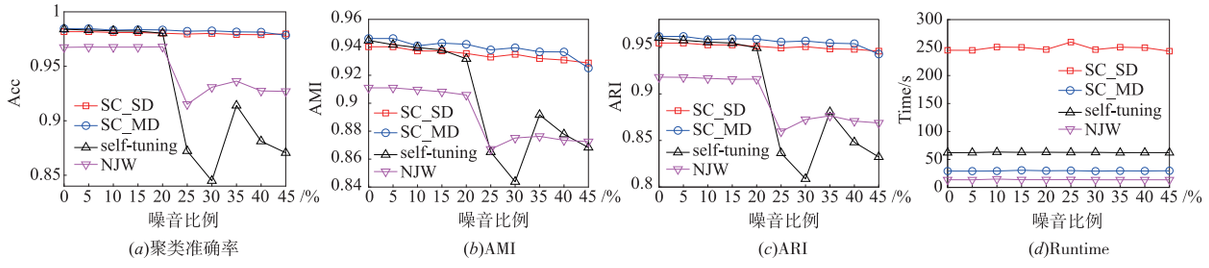

图1 SC_SD、SC_MD、NJW和self-tuning算法鲁棒性测试实验结果比较

图 1(a)、(b)和(c)的实验结果揭示:提出的 SC_SD 和 SC_MD 算法对噪音具有非常强的鲁棒性,其聚类准确率、聚类结果指标 AMI 和 ARI 均优于传统谱聚类算法 NJW 和自适应的谱聚类算法 self-tuning. SC_SD 和 SC_MD 的聚类性能相比,SC_MD 的聚类准确率、聚类结果指标 AMI 和 ARI 略优于 SC_SD 算法. 传统谱聚类算法 NJW 和自适应谱聚类算法 self-tuning 均受到噪音影响,且在噪音比例超过 20% 时,其聚类结果急剧变差,self-tuning 算法的聚类性能受影响更大,其聚类结果甚至变得不如 NJW 算法. 由此可见,self-tuning 算法受噪音影响很大.

图 1(d)的运行时间比较揭示,各算法的收敛速度不受噪音影响. 传统谱聚类算法 NJW 的收敛速度最快,提出的 SC_MD 算法仅次于 NJW, self-tuning 算法的收敛速度大约比 SC_MD 算法慢一倍,提出的 SC_SD 算法的收敛速度最慢.

表 13 实验结果揭示:提出的 SC_SD 和 SC_MD 算法,在不需要任何先验知识的情况下,取得了非常好的聚类效果,特别是 SC_MD 算法的聚类效果是 SC_SD、SC_MD、NJW + 和 self-tuning + 谱聚类算法中最优的,这与图 1 所示的 SC_SD、SC_MD、NJW 和 self-tuning 在 10 组不同比例噪音人工模拟数据集的实验结果一致,说明尺度参数的重要性,也说明提出的 SC_SD 和 SC_MD 的优越性.

表 13 SC_SD、SC_MD、NJW + 和 self-tuning + 在表 12 参数生成的不含噪音数据集的实验结果

算法	参数	Acc	AMI	ARI
NJW +	$\sigma = 1.5$	0.9466	0.8707	0.8653
	$\sigma = 1$	0.9496	0.8762	0.8729
	$\sigma = 0.5$	0.9584	0.8931	0.8952
	$\sigma = 0.35$	0.9686	0.9134	0.9211
self-tuning +	$p = 5$	0.9814	0.9400	0.9536
	$p = 10$	0.7568	0.7644	0.7083
	$p = 15$	0.9834	0.9434	0.9587
	$p = 20$	0.9840	0.9449	0.9601
SC_SD	–	0.9818	0.9404	0.9546
SC_MD	–	**0.9846**	**0.9463**	**0.9617**

4.4 伸缩性测试实验

为了测试提出的 SC_SD 和 SC_MD 算法是否具有伸缩性,根据表 12 参数生成 7 组规模分别为 1000 * 5 = 5000, 1500 * 5 = 7500, 2000 * 5 = 10000, 2500 * 5 = 12500, 3000 * 5 = 15000, 3500 * 5 = 17500, 4000 * 5 = 20000 的不含噪音的人工数据集. 在这 7 组数据集分别

运行提出的 SC_SD 和 SC_MD 算法,以及传统谱聚类算法 NJW 和自适应谱聚类算法 self-tuning,NJW 的尺度参数 $\sigma=0.35$,self-tuning 的尺度参数 $p=20$,该参数由 4.3 节的鲁棒性实验获得. 各算法的聚类准确率、聚类结果指标 AMI 和 ARI,以及运行时间比较如图 2 所示.

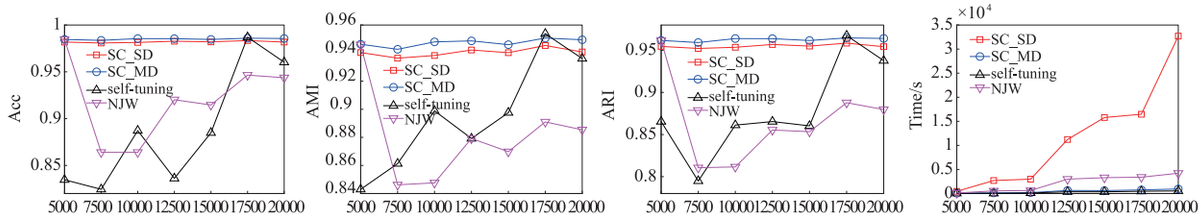

图2　SC_SD、SC_MD、NJW和self-tuning算法伸缩性实验结果比较

图 2(a)、(b)和(c)所示的各算法的聚类准确率、聚类结果指标 AMI 和 ARI 实验结果比较显示,提出 SC_SD 和 SC_MD 算法的聚类效果非常好,且 SC_MD 算法在各数据集的聚类性能略优于 SC_SD 算法,但是传统谱聚类算法 NJW 和自适应谱聚类算法 self-tuning 在不同规模数据集的聚类效果很不稳定,尤其 self-tuning 算法非常不稳定. 由此可见,提出 SC_SD 和 SC_MD 算法的性能优于传统谱聚类算法 NJW 与自适应的谱聚类算法 self-tuning. 分析原因在于:提出的 SC_SD 和 SC_MD 算法是完全自适应的谱聚类算法,它们能根据样本分布信息,自适应地找到合适的局部尺度参数,且在聚类阶段使用确定的 SD_K-medoids 算法代替了传统谱聚类算法 NJW 和自适应谱聚类算法 self-tuning 使用的 K-means 算法,从而使得 SC_SD 和 SC_MD 算法的聚类结果更接近数据集样本的实际分布. NJW 和 self-tuning 算法的人工尺度参数、聚类阶段的 K-means 均增加了聚类结果的不稳定,而且尺度参数对聚类结果的影响,随数据集规模增大而增加.

图 2(d)关于各算法聚类时间的比较显示,SC_MD 算法和 NJW 算法需要的聚类时间最少,且聚类时间(收敛速度)几乎不受数据集规模影响. self-tuning 的聚类时间在数据集规模超过 10000 时,有明显增加. 提出的 SC_SD 算法的聚类时间随着数据集规模增加而增长. 由此可见,提出的 SC_MD 算法和传统谱聚类算法 NJW 的伸缩性最好,可被用于大数据聚类,但是 NJW 算法的聚类结果严重依赖于其尺度参数 σ,且聚类性能远不如提出的完全自适应的谱聚类算法 SC_MD.

综合图 2 实验结果分析可见,提出的完全自适应的谱聚类算法 SC_SD 和 SC_MD 的聚类效果绝对优于传统谱聚类算法 NJW 和自适应谱聚类算法 self-tuning,能完全自适应地发现不同规模数据集的真实分布,尤其是 SC_MD 算法具有非常好的伸缩性,完全可用于较大规模数据集的聚类分析.

5　结论

提出了两种完全自适应的谱聚类算法 SC_SD 和 SC_MD,分别以样本 i 标准差 std_i 及样本 i 到所有样本的距离均值 d_{mean_i} 为邻域半径,以样本 i 的邻域标准差为其自适应的局部尺度参数,充分利用了数据的分布信息,解决了传统谱聚类算法 NJW 的尺度参数需要人为设置,自适应谱聚类算法 self-tuning 的局部尺度参数会受到离群点影响且需要人为给定的缺陷;同时以 SD_K-medoids 算法代替谱聚类算法中的 K-means 算法,彻底解决了谱聚类算法的聚类结果不稳定问题.

UCI 真实数据集和人工数据集实验测试,以及鲁棒性和伸缩性实验测试表明,提出的谱聚类算法 SC_SD 和 SC_MD 具有如下优势:(1)聚类性能优于传统谱聚类算法 NJW 和自适应谱聚类算法 self-tuning;(2)对噪音数据具有很好的鲁棒性,聚类效果和聚类时间不受噪音影响;(3)具有很好的伸缩性,聚类效果不受数据集规模影响;(4)SC_MD 算法不仅聚类效果好且聚类时间几乎不受数据集规模影响,适于发现较大规模数据集的真实分布.

参考文献

[1] Xie J Y, Jiang S, Xie W X, et al. An efficient global K-means clustering algorithm[J]. JCP, 2011, 6(2): 271-279.

[2] Han J W, Kamber M. Data Mining: Concepts and Techniques[M]. Beijing: China Machine Press, 2006. 83-84.

[3] Sajana T, Rani C M S, Narayana K V. A survey on clustering techniques for big data mining[J]. Indian Journal of Science and Technology, 2016, 9(3): 1-12.

[4] Rai P, Dwivedi R K. Clustering techniques for unsupervised Learning[J]. International Journal of Management, IT and Engineering, 2012, 2(11): 462-571.

[5] Macqueen J. Some methods for classification and analysis

of multivariate observations[A]. Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability[C]. Berkeley: University of California Press, 1967. 281 – 297.

[6] Fu G Y. Optimization methods for fuzzy clustering[J]. Fuzzy Sets and Systems, 1998, 93(3): 301 – 309.

[7] Kaufman L, Rousseeuw P J. Finding Groups in Data: An Introduction to Cluster Analysis[M]. New York, USA: John Wiley & Sons, 1990. 126 – 163.

[8] Luxburg U V. A tutorial on spectral clustering[J]. Statistics and Computing, 2007, 17(4): 395 – 416.

[9] Perona P, Freenman W. A factorization approach to grouping[A]. European Conference on Computer Vision[C]. Berlin: Springer, 1998. 655 – 670.

[10] Shi J, Malik J. Normalized cuts and image segmentation[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000, 22(8): 888 – 905.

[11] Ng A Y, Jordan M I, Weiss Y. On spectral clustering: analysis and an algorithm[A]. Advances in Neural Information Processing Systems (NIPS14)[C]. Cambridge MA: MIT Press, 2002. 849 – 856.

[12] Zelnik-Manor L, Perona P. Self-tuning spectral clustering[A]. Advances in Neural Information Processing Systems (NIPS17)[C]. Cambridge MA: MIT Press, 2005. 1601 – 1608.

[13] 王玲, 薄列峰, 焦李成. 密度敏感的谱聚类[J]. 电子学报, 2007, 35(8): 1577 – 1581.
WANG Ling, BO Lie-feng, JIAO Li-cheng. Density sesitive spectral clustering[J]. Acta Electronica Sinica, 2007, 35(8): 1577 – 1581. (in Chinese)

[14] 王娜, 李霞. 基于监督信息特性的主动半监督谱聚类算法[J]. 电子学报, 2010, 38(1): 172 – 176.
WANG Na, LI Xia. Activesemi-supervised spectral clustering based on pairwise constraints[J]. Acta Electronica Sinica, 2010, 38(1): 172 – 176. (in Chinese)

[15] Xie J Y, Zhou Y, Ding L J. Local standard deviation spectral clustering[A]. 2018 IEEE International Conference on Big Data and Smart Computing (BigComp)[C]. Piscataway: IEEE, 2018. 242 – 250.

[16] 谢娟英, 高瑞. 方差优化初始中心的 K-medoids 聚类算法[J]. 计算机科学与探索, 2015, 9(8): 973 – 984.
Xie Juanying, Gao Rui. K-medoids clustering algorithms with optimized initial seeds by variance[J]. Journal of Frontiers of Computer Science and Technology, 2015, 9(8): 973 – 984. (in Chinese)

[17] Blake C, Merz C J. UCI Repository of machine learning databases[DB/OL]. http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998-04-02.

[18] Vinh N X, Epps J, Bailey J. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance[J]. Journal of Machine Learning Research, 2010, 11(Oct): 2837 – 2854.

[19] Hubert L, Arabie P. Comparing partitions[J]. Journal of classification, 1985, 2(1): 193 – 218.

[20] 卜东波, 白硕. 聚类分类中的粒度原理[J]. 计算机学报, 2002, 25(8): 810 – 816.
Piao Dongbo, Bai Shuo. Principle of granularity in clustering and classification[J]. Chinese Journal of Computers, 2002, 25(8): 810 – 816. (in Chinese)

[21] Chang H, Yeung D Y. Robust path-based spectral clustering[J]. Pattern Recognition, 2008, 41(1): 191 – 203.

[22] Fränti P, Virmajoki O. Iterative shrinking method for clustering problems[J]. Pattern Recognition, 2006, 39(5): 761 – 775.

作者简介

谢娟英 女, 1971 生于陕西西安, 陕西师范大学计算机科学学院教授, 主要研究领域为机器学习、数据挖掘和生物医学大数据分析.
E-mail: xiejuany@snnu.edu.cn

丁丽娟 女, 1994 生于陕西安康, 陕西师范大学计算机科学学院硕士研究生, 主要研究领域为机器学习、数据挖掘.
E-mail: lijuan_ding@163.com