计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.200100112

基于自然最近邻的密度峰值聚类算法

汤鑫瑶　张正军　储　杰　严　涛
南京理工大学理学院　南京 210094
(1015562103@qq.com)

摘　要　针对密度峰值聚类算法(Density Peaks Clustering,DPC)需要人为指定截断距离d_c,以及局部密度定义简单和一步分配策略导致算法在复杂数据集上表现不佳的问题,提出了一种基于自然最近邻的密度峰值聚类算法(Density Peaks Clustering based on Natural Nearest Neighbor,NNN-DPC)。该算法无需指定任何参数,是一种非参数的聚类方法。该算法首先根据自然最近邻的定义,给出新的局部密度计算方法来描述数据的分布,揭示内在的联系;然后设计了两步分配策略来进行样本点的划分。最后定义了簇间相似度并提出了新的簇合并规则进行簇的合并,从而得到最终聚类结果。实验结果表明,在无需参数的情况下,NNN-DPC 算法在各类数据集上都有优秀的泛化能力,对于流形数据或簇间密度差异大的数据能更加准确地识别聚类数目和分配样本点。与 DPC、FKNN-DPC(Fuzzy Weighted K-nearest Density Peak Clustering)以及其他 3 种经典聚类算法的性能指标相比,NNN-DPC 算法更具优势。

关键词:聚类算法;自然最近邻居;密度峰值;局部密度
中图法分类号　TP301.6

Density Peaks Clustering Algorithm Based on Natural Nearest Neighbor

TANG Xin-yao,ZHANG Zheng-jun,CHU Jie and YAN Tao
School of Science,Nanjing University of Science and Technology,Nanjing 210094,China

Abstract　Aiming at the problem that the density peak clustering (DPC) algorithm requires manually selected parameters (cutoff distance d_c),as well as the problem of a poor performance on complex data sets caused by the simple definition of local density and the one-step assignment strategy,a new density peak clustering algorithm based on natural nearest neighbors (NNN-DPC) is proposed. The algorithm does not need to specify any parameters and is a non-parametric clustering method. Based on the definition of natural nearest neighbors,this algorithm firstly gives a new local density calculation formula to describe the distribution of data,and reveals the internal connection. A two-step assignment strategy is designed to divide the sample points. Finally,the similarity between clusters is defined,and a new cluster merging rule is proposed to merge the clusters to obtain the final clustering result. The experimental results show that without parameters,the NNN-DPC algorithm has excellent generalization ability on various types of data sets,and can more accurately identify the number and distribution of clusters on manifold data or data with large differences of density between clusters,and assign sample points to the corresponding clusters. Compared with the performance indicators of DPC,FKNN-DPC,and three other classic clustering algorithms,the NNN-DPC algorithm has a great advantage.

Keywords　Clustering algorithm,Natural nearest neighbor,Density peaks,Local density

1　引言

聚类分析是一种无监督的机器学习算法,它按照相似性将一组给定的数据点分配到不同的集合,使得同一个集合内的样本点的相似性较高,而不同集合间的样本点的相似度较低,这些集合又被称为簇。

在聚类方法的发展中,涌现出了很多经典算法,典型的划分算法有 K-means[1]、K-medoids[2]、近邻传播(AP)算法[3],以及基于密度的 DBSCAN 算法[4],这些算法因为计算简单而被广为使用,但均有较大的局限性。

2014 年,Alex 等[5]提出了一种新型的基于密度峰值的聚类算法 CFSFDP(Clustering by Fast Search and Find of Density Peaks,简称 DPC),该方法不仅能够快速识别聚类数目,找到聚类中心,而且对非球形簇有很好的聚类效果。尽管 DPC 算法具有计算简单、快速的优势,但仍然存在一些不足:1)该算法对截断距离d_c的选取较敏感,难以估计合理的截断距离,

到稿日期:2020-01-07　返修日期:2020-06-09　本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家自然科学基金(11671205)
This work was supported by the National Natural Science Foundation of China(11671205).
通信作者:张正军(zzjnj@163.com)

并且全局只有一个截断距离，当数据同时存在高密度簇和低密度簇时，易导致高密度簇被错误地分为多个簇。2）一步分配策略容易导致误差不断扩大。近年来，很多针对 DPC 算法的改进算法被提出。Yang 等[6]提出了利用信息熵来计算截断距离d_c，使得算法更容易在决策图中挑选出聚类中心。Wang 等[7]提出利用基尼不纯度找到最优的d_c值。Liu 等[8]提出了基于共享最近邻居的局部密度度量和距离 δ 的自适应度量，在此基础上提出了新的快速密度峰值算法 SNN-DPC（shared-nearest-neighbor-based clustering by fast search and find of density peaks），其在各类数据集上的聚类性能明显优于 DPC。此外还有很多基于 K 近邻（KNN）的方法不断被提出。Du 等[9]提出了基于 K 近邻的 DPC-KNN 算法（Density Peaks Clustering Based on K Nearest Neighbor），引入 KNN 的概念定义新的局部密度度量，避免了截断距离d_c的计算。Xie 等[10]提出了模糊加权 K-近邻（FKNN-DPC）的密度峰值聚类算法，该算法设计了新的密度峰值搜索方法和两步分配策略，解决了 DPC 算法中密度测量的不均匀性和一步分配导致的误差传递问题，提高了聚类质量。但是，DPC-KNN 和 FKNN-DPC 等基于 K 近邻的算法的聚类结果依赖于参数 K 的选取，不同的 K 值对最终的结果有很大影响。

针对上述参数敏感的问题，本文提出了基于自然最近邻[11-13]的密度峰值算法（NNN-DPC）。该算法根据自然最近邻的定义，从数据中自动获得每个样本点的近邻数，并以此计算每个样本点的局部密度和样本点与较高密度点间的距离。在分配过程中，本文提出了两步分配策略，分配结束后引入了簇间相似性的概念来合并相似性较高的簇，从而获得最终的聚类结果。实验结果表明，本文的 NNN-DPC 算法在无需参数的情况下，获得了比 DPC 算法更加优秀的结果。

2 DPC 算法与分析

2.1 DPC 算法

DPC 算法主要遵循两个思想：1）聚类中心本身的密度比它的邻居更大；2）聚类中心与具有更高密度样本点的距离相对较大。设有数据集 $X=\{x_1, x_2, \cdots, x_N\}$，$x_i=(x_{i1}, x_{i2}, \cdots, x_{ip})^{\mathrm{T}}$，$i=1,2,\cdots,N$，$x_{ij}$表示第 i 个数据点的第 j 维属性值。DPC 算法只需要计算两个变量：样本点的局部密度 ρ 和该点与密度更高的最近邻之间的距离 δ。对于每个数据点x_i，$i=1,2,\cdots,N$，局部密度ρ_i的计算式为：

$$\rho_i=\sum_{j,j\neq i}\chi(d_{ij}-d_c) \tag{1}$$

其中，χ 是一个示性函数，若$d_{ij}<d_c$则取 1，否则取 0；$d_{ij}=\mathrm{dist}(x_i,x_j)$表示两个数据点之间的某种距离，一般为欧氏距离；$d_c$被称为截断距离。可以认为，局部密度$\rho_i$等价于与点$x_i$的距离小于$d_c$的点的数目。

为了避免出现大量相同的局部密度值，一般在 DPC 算法中使用高斯核函数来计算局部密度 ρ：

$$\rho_i=\sum_{j,j\neq i}\exp\left(-\left(\frac{d_{ij}}{d_c}\right)^2\right) \tag{2}$$

截断距离d_c是式(1)和式(2)中唯一的变量，在原算法中，d_c的取值方法如下：

$$d_c=d_{N_d*p} \tag{3}$$

其中，$N_d=C_N^2$表示样本配对的数量，$d_{N_d*p}\in D=[d_1,d_2,\cdots,d_{N_d}]$；$D$ 是两两样本间距离的集合，并且按照升序排列；N 表示样本容量；d_{N_d*p}表示集合 D 中第N_d*p个距离，原算法中 p 取 1%～2%。

点x_i的距离δ_i定义为：

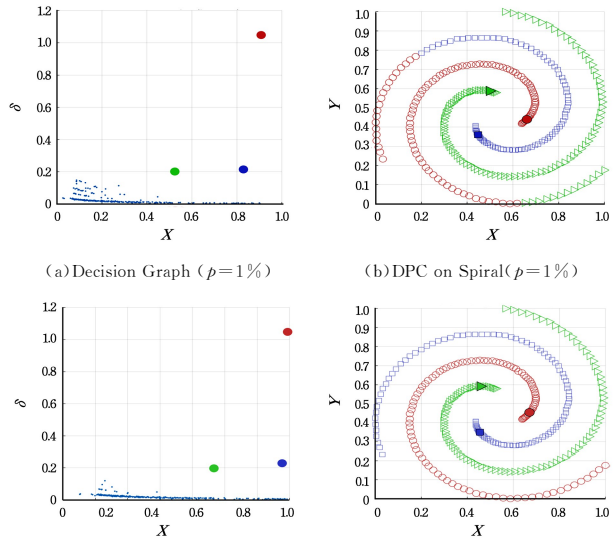
$$\delta_i=\begin{cases}\min\limits_{j:\rho_j>\rho_i}(d_{ij}), & \rho_i<\max(\rho)\\ \max\limits_{j}(d_{ij}), & \rho_i=\max(\rho)\end{cases} \tag{4}$$

即对于非密度最高的样本点，计算点 i 与较高密度最近邻的距离。而对于最高密度的点，计算点 i 和最远的点之间的距离。

为了更好地确定聚类中心的个数，DPC 算法还需要计算各样本点的 γ 值：

$$\gamma_i=\rho_i\delta_i \tag{5}$$

在计算出所有点的局部密度ρ_i和距离δ_i之后，以局部密度ρ_i为横轴，距离δ_i为纵轴，绘制决策图（见图 1(a)和图 1(c)）。决策图中ρ_i和δ_i均相对较大的点（或者 γ 值相对较大的点）将被选择为聚类中心，剩下的点被归于密度较高的最近邻所属的簇，进而得到最终的聚类结果。



(a) Decision Graph ($p=1\%$)　(b) DPC on Spiral($p=1\%$)

(c) Decision Graph($p=2\%$)　(d) DPC on Spiral($p=2\%$)

图 1　当 $p=1\%$和 $p=2\%$时 DPC 在 Spiral 数据集上的结果

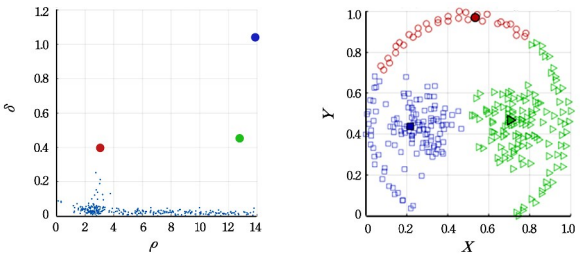
Fig. 1　Results of DPC on Spiral with $p=1\%$ and $p=2\%$

2.2 DPC 算法的缺陷

本文主要针对 DPC 算法的以下两种缺陷进行研究。

（1）参数敏感问题。DPC 中截断距离d_c按照比例 p 从样本点对的距离中取值，不同的 p 值将对最终的聚类结果产生很大的影响。如图 1 所示，4 幅子图分别为 DPC 算法在 Spiral 合成数据集上的决策图和聚类结果，同种颜色和形状的点表示同一簇元素，p 分别取 1%和 2%。可以看到，两种情况下 DPC 算法均从决策图中识别了正确的聚类数和聚类中心，但是 DPC 只有在 $p=2\%$时得到了正确的聚类结果。

（2）一步分配所带来的误差传递问题。在 Pathbased 数据集上，DPC 算法从决策图中正确地识别出了 3 个聚类中心（见图 2(a)），但是外围流形簇的部分数据点被错误地分配到了两个块状簇中（见图 2(b)）。



(a)Decision Graph (p=2%) (b)DPC on Pathbased(p=2%)

图 2 当 p=2%时 DPC 在 Pathbased 数据集上的结果

Fig. 2 Results of DPC on Pathbased when p= 2%

3 NNN-DPC 算法

3.1 自然最近邻

在对数据结构性质的研究中，最近邻居的概念被提出，其中常用的近邻方法是 Stevens 提出的 K-邻域和 ε-邻域[14]。

自然最近邻居（Natural Nearest Neighbor，NNN）是 Zou[15] 于 2011 年提出的一种新的邻居定义，与 K-近邻和 ε-近邻相比，其最大的不同之处在于它是由数据结构自身生成的，无需任何参数。它的主要思想是，若数据点 a 出现在点 b 的 r-邻域内，则点 b 为点 a 的自然最近邻。

定义 1(自然最近邻居) 对于数据点 x，称数据点 y 为数据点 x 的自然最近邻居，若 x 在 y 的 r-邻域内，且对于数据集中其他任意点 z，都至少存在一个数据点的 r-邻域包含 z。

因此，数据集中密集区域的数据点往往具有更多的自然邻居，稀疏区域的数据点拥有较少的自然邻居。

定义 2(自然特征值 $supk$) 数据集的自然特征值为使任意的数据点 x 都至少被另一个数据点 $y(y\neq x)$ 的 r-邻域包含的最小 r 值，其数学定义如下：

$$supk=\min\{r|\ \forall x\in X,\exists y\in X,y\neq x,\text{s.t.}\ x\in NN_r(y)\} \tag{6}$$

其中，$NN_r(y)$ 表示数据点 y 的 r-邻域，即距离点 y 最近的 r 个点的集合。$supk$ 也被称为平均自然邻居数[15]。

显然，当存在离群点时，为使离群点也有自然最近邻居，$supk$ 值会过大。在使用 k-d 树搜索邻居的情况下，自然最近邻居的原算法的时间复杂度为 $O(supk*N\log(N))$，当 $supk$ 值很大(极端情况下 $supk$ 为 $N-1$)时，会大大增加算法的时间复杂度。因此，Huang[16] 提出了优化的自然最近邻搜索算法，其主要的改进思想为：随着 r 值的增加，若自然邻居数为 0 的数据点数量不变，则停止搜索。根据这一思想，自然最近邻搜索算法的伪代码如算法 1 所示。

算法 1 NNN-Searching

输入：数据集 X（维数为 N * p）

输出：supk，各点的自然最近邻集合 NNN={NNN(1)，NNN(2)，…，NNN(N)}，各点自然最近邻的数量 nb={nb(1)，nb(2)，…，nb(N)}

```
1. 初始化：对数据集 X 进行归一化
2. 初始化：r=0，flag=0，before=N，after=0；
3. 初始化：NNN(i)=Ø，nb(i)=0，i=1，2，…，N；
4. while flag==0 do
5.    r=r+1；
6.    for each point i in X
7.    using k-d tree find the r_th neighbor of i：nn_r(i)；
8.    nb(nn_r(i))=nb(nn_r(i))+1；
9.       NNN(nn_r(i))=NNN(nn_r(i))∪{i}；
10.   end for
11.   if r≥2
12.         before=after；
13.   end if
14. after=count(nb==0)；
15.   if before==before
16.   flag=1；
17. end if
18. end while
19. supk=r.
```

3.2 NNN-DPC 算法

定义 3(数据点的局部密度) 根据自然最近邻居的定义以及算法 1，本文将 NNN-DPC 算法的局部密度定义为：

$$k(i)=\min\{supk,nb(i)\} \tag{7}$$

$$\rho_i=\begin{cases}\sum\limits_{\substack{j\in NNN(i)\\ d_{ij}\leqslant dk_{(i)}}}\exp(-d_{ij}), & nb(i)>0\\ 0, & nb(i)=0\end{cases} \tag{8}$$

其中，$nb(i)$ 是点 i 的自然最近邻居数，d_{ij} 是点 i 和点 j 之间的欧氏距离，$NNN(i)$ 表示点 i 的自然最近邻域，$k(i)$ 为 $supk$ 和 $nb(i)$ 中的较小值，$dk_{(i)}$ 表示点 i 与自然最近邻集合中第 $k(i)$ 近的点之间的欧氏距离。式(8)只利用了较近的几个自然最近邻信息，更容易在决策图中识别出低密度簇的簇心。并且该局部密度是一种无参数度量。

本文算法中，距离较高密度最近邻的距离 δ 的定义沿用式(4)，依然使用决策图作为聚类中心选择的依据，以 γ 值为辅助，挑选出 ρ 和 δ 都较大的点作为簇心。

定义 4(离群点) 离群点由于距离正常的簇较远，很难被其他数据点认为是近邻，因此由自然最近邻的定义和算法 1 可知，自然最近邻数为 0，即 $nb(i)=0$ 的数据点 i 可以自然地被认为是离群点(噪声)。

定义 5(样本间相似度) 对于数据点 i 和 $j(i\neq j$ 且均为非离群点)，本文使用式(12)来衡量两者之间的相似性：

$$ave\,d_i=\frac{1}{nb(i)}\sum_{m\in NNN(i)}d_{im} \tag{9}$$

$$\alpha_{ij}=\min\left\{\frac{ave\,d_i}{ave\,d_j},\frac{ave\,d_j}{ave\,d_i}\right\} \tag{10}$$

$$inter(i,j)=\{NNN(i)\cap NNN(j)\} \tag{11}$$

$$sim(i,j)=\alpha(i,j)*\frac{|inter(i,j)|+1}{d_{ij}} \tag{12}$$

离群点与其他任何点的相似性均为 0。

$ave\,d_i$ 是数据点 i 与它自然最近邻的平均距离，α_{ij} 为缩放系数，$inter(i,j)$ 为数据点 i 和 j 的自然最近邻集合的交集，常数 1 为调节常数，避免没有自然最近邻交集的两点相似度为 0，增强了相似性度量的稳健性，也使得 NNN-DPC 算法中非簇心点的两步分配正常进行。d_{ij} 为两点之间的欧氏距离。

本文将样本 i 对簇 c 的隶属度定义为：

$$P_i^c=\sum_{j\in NNN(i),y_j=c}\omega(i,j)sim(i,j) \tag{13}$$

其中，$\omega(i,j)=sim(i,j)/\sum\limits_{l\in NNN(i)}sim(i,l)$ 为权重，y_j 是数据点

j 的簇标记。式(13)的权重表明,本文的 NNN-DPC 算法更偏向于将未分配的点归到相似性高的点所属的簇,而不是简单的多数投票规则。

定义 6(相似可达) 若样本点 j 为点 i 的自然最近邻,且 j 与 i 的相似度为 i 与其自然最近邻相似度中最高的相似度,即若 $j\in NNN(i)$,$sim(i,j)=\max\limits_{m\in NNN(i)}[sim(i,m)]$,则称点 j 由点 i 相似可达。

定义 7(簇核心区) 对于一个未分配的聚类中心 i,其自然最近邻居集合为 $NNN(i)$,将点 i、$NNN(i)$以及由 $NNN(i)$出发、由相似可达概念经过的点统称为该簇的簇核心区。

在得到初始簇心、相似度矩阵后,本文为 NNN-DPC 算法设计了两步分配策略,其主要思想为:1)将聚类中心按照局部密度降序排列,不断挑选出未分配的聚类中心,分配簇标记,并得到相应的簇核心区;2)按照式(13)定义的簇隶属度将未分配的非离群点分配到隶属度最高的簇。

定义 8(簇间相似度) 假设有簇 C_p 和簇 C_q,两个簇中互为自然最近邻居的点对数量为 $DN(C_p,C_q)$,这两个簇所有样本点的平均自然最近邻数分别为 $mnb(C_p)$和 $mnb(C_q)$。称式(14)定义的 $S(C_p,C_q)$值为簇 C_p 和簇 C_q 的簇间相似度:

$$S(C_p,C_q)=\frac{DN(C_p,C_q)}{mnb(C_p)p_1+mnb(C_q)(1-p_1)} \tag{14}$$

其中,$p_1=|C_p|/(|C_p|+|C_q|)$,$|C_p|$ 和 $|C_q|$ 分别表示两个簇的样本数量。式(14)的分母表示两个簇的平均自然最近邻数。在 NNN-DPC 算法中,将 $S(C_p,C_q)\geqslant 1$ 的簇合并,即当两个簇中互为自然最近邻居的点对数量高于这两个簇的平均自然最近邻数时,将这两个簇合并。

3.3 算法步骤

NNN-DPC 算法的步骤如算法 2 所示。

算法 2 NNN-DPC

输入:数据集 X

输出:聚类结果 C

初始化:归一化数据集 X,任意的数据点 i(i=1,2,…,N),簇标签 cl(i)=−1,访问标记 visit(i)=0。

步骤 1 使用算法 1 获得每个数据点的自然最近邻集合 NNN(i)和自然邻居数 nb(i)。

步骤 2 使用式(8)和式(4)计算每个点的局部密度 ρ 和距离 δ。

步骤 3 根据决策图挑选聚类中心,数量记为 NC′,簇标记集合 c={1,2,…,NC′}。

步骤 4 将自然最近邻数为 0 的点(例如点 z)记作离群点,并标记为已访问,即 visit(z)=1。

步骤 5 初步分配:

步骤 5.1 从未被访问的聚类中心内选取局部密度最大的点 i,赋予未使用的簇标记 $c_i\in c$。将点 i 的自然最近邻也赋予簇标记 c_i,将这些点标记为已访问,即 cl(i)=cl(NNN(i))=c_i,visit(i)=visit(NNN(i))=1。

步骤 5.2 将步骤 5.1 中的自然最近邻集合 NNN(i)放入序列 Q 中。

步骤 5.3 从 Q 中取出第一个数据点 p,并将其从 Q 中删除,在 p 的自然最近邻域 NNN(p)中挑选满足 $sim(p,q)=\max\limits_{j\in NNN(p)}(sim(p,j))$的点 q,即点 q 是点 p 最相似的自然最近邻。

步骤 5.4 若点 q 未被访问,则将点 q 归到点 p 所属的簇,并将点 q 添加到序列 Q 的末尾,将其标记为已访问,即 visit(q)=1;否则返回步骤 5.3。

步骤 5.5 若序列 Q 非空,则返回步骤 5.3。

步骤 5.6 若还有未访问的簇心,则返回步骤 5.1。

步骤 6 二步分配。假设初步分配得到了 NC″个簇核心区:

步骤 6.1 对经过上述步骤后仍未被访问的数据点 j,用式(13)计算其对各个簇的隶属度 P_j^c。

步骤 6.2 计算每个点 j 的最大隶属度 MP(j)=max{$P_j^c|c=1,2,\cdots,NC''$}以及最大隶属度所属的簇 MC(j)=$\underset{c}{\text{argmax}}\{P_j^c|c=1,2,\cdots,NC''\}$。

步骤 6.3 分配具有最大隶属度的点,即若 MP(p)=$\max\limits_{j}$(MP(j))>0,则将点 p 分配到簇 MC(p)中,将点 p 标记为已访问,即 visit(p)=1。

步骤 6.4 更新 MP(p)=0。对于自然最近邻中包含点 p 的点 q,即 $\forall q\in\{q|p\in NNN(q)\}$,更新 $P_q^{MC(p)}=P_q^{MC(p)}+\omega(q,p)*sim(q,p)$,其中 ω(q,p)为式(13)中定义的权重。更新 MP(q)=max{$P_q^c|c=1,2,\cdots,NC''$}和 MC(q)=$\underset{c}{\text{argmax}}\{P_q^c|c=1,2,\cdots,NC''\}$。

步骤 6.5 如果所有点都被访问,则结束分配,否则返回步骤 6.3。

步骤 7 由式(14)计算簇与簇之间的 S 值,若 S≥1,则合并相应的两个簇,并返回最终的聚类结果,假设最终有 NC 个簇,将聚类结果记为 C={$C_1,C_2,\cdots,C_{NC}$}。其中,C_i(i=1,2,…,NC)表示有相同簇标记的样本点集合。

3.4 复杂度分析

(1)空间复杂度。存储距离矩阵的空间复杂度为 $O(N^2)$,除此之外,储存相似度矩阵的空间复杂度为 $O(N^2)$,储存分配概率矩阵的空间复杂度为 $O(M^2)$,$M<N$,自然最近邻集合平均空间复杂度为 $O(N*supk)$。因此,NNN-DPC 算法的总体空间复杂度为 $O(N^2)$,与 DPC 算法相同。

(2)时间复杂度。NNN-DPC 算法的时间复杂度取决于以下几个方面:1)计算距离矩阵,其时间复杂度为 $O(N^2)$;2)NNN-searching 算法,因为使用了 *k-d* 树搜索,搜索 i 的第 r 个近邻时间复杂度为 $O(\log(N))$,因此时间复杂度为 $O(supk*N*\log(N))$;3)计算局部密度 ρ,由于已有自然最近邻集合,故时间复杂度为 $O(N*supk)$;4)计算距离 δ,其时间复杂度为 $(O(N^2))$;5)计算相似度矩阵,其时间复杂度为 $O((N^2*supk^2)$;6)将点分配到相应的簇中,第一步分配的时间复杂度不超过 $O(n_1*supk*NC')$,n_1 为非离群点的数目,NC'为初始聚类中心的数目,第二步分配是将剩余的点(假设有 n_2 个)分配给最可能的集群,时间复杂度为 $O(n_2^2)$。因此,总的时间复杂度为 $O((N^2*supk^2)$,稍高于 DPC 算法。

4 实验结果与分析

本文选取了 6 个合成数据集和 6 个真实数据集进行实验。本节选取了经典的 *K*-means、DBSCAN、AP 算法、Alex 提出的 DPC 算法,以及改进的 FKNN-DPC 算法进行对比。

本文采用了 3 个聚类性能指标:调整互信息(Adjusted Mutual Information,AMI)[17]、调整兰德系数(Adjusted Rand Index,ARI)[17] 和 Fowlkes-Mallows 指数(Fowlkes-Mallows

index，FMI)[18]。其中，AMI 和 FMI 的取值范围均为[0，1]，ARI 的取值范围为[－1，1]，三者的值越接近 1，聚类效果就越好。

编程环境为 Win10，NNN-DPC 算法、KNN-DPC 算法以及 DPC 算法使用 Matlab 2018a 进行编程实验，AP 算法、DBSCAN 算法、*K*-means 算法使用 Python 的 sklearnlibrary[19]进行实验。

实验数据均不包含离群点，为了公平、客观地进行对比实验，凸显 NNN-DPC 算法在无需参数的情况下依然可以获得更好的效果，本节对上述算法做了如下调整和说明：1）在 NNN-DPC 算法中，将算法定义的离群点分配到欧氏距离最近的点所属的簇中；2）去除 DPC 算法的离群点识别部分，且算法的比例参数 p 设置为 2%；3）*K*-means 算法中参数 K 取正确聚类数；4）FKNN-DPC，AP 和 DBSCAN 在实验中取局部最优的结果，其中 FKNN-DPC 中 参数 $K\in\{3,6,9,5,10,15\}$，DBSCAN 中参数 $\varepsilon\in\{0.04,0.08,0.2,0.6\}$，$MinPts\in\{4,6,8,12,20,30\}$，AP 算法的参数为程序中的默认参数。在进行聚类任务前均对数据进行了归一化处理。

4.1 合成数据集实验

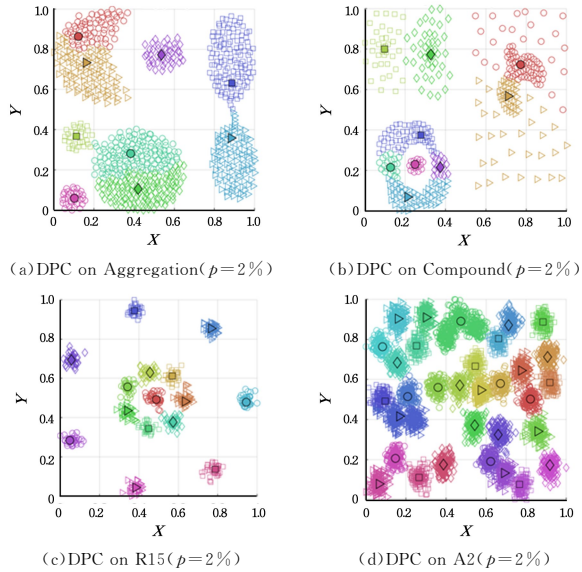
本节选取 6 个合成数据集进行实验，这些数据集的样本量、维数等详细信息如表 1 所列。

图 1(d)、图 2(b)以及图 3(a)－图 3(d)分别显示了 DPC 算法($p=2\%$)在这 6 个合成数据集上的聚类结果，图 4(a)－图 4(f)分别显示了 NNN-DPC 算法在合成数据集上的最终聚类结果。其中，不同的颜色和形状代表不同的簇，黑框实色图形标记的点为决策图中选出的聚类中心。

表 1 合成数据集

Table 1 Synthetic dataset

数据集	N	D	Classes	Source
Spiral	312	2	3	文献[20]
Pathbased	300	2	2	文献[20]
Aggregation	788	2	7	文献[21]
Compound	399	2	6	文献[21]
R15	600	2	15	文献[22]
A2	5250	2	35	文献[23]

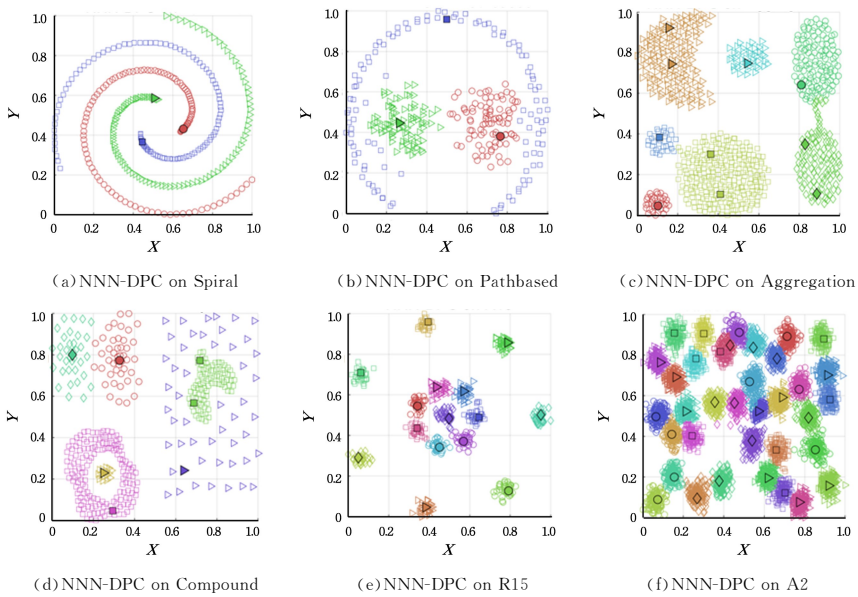


(a) DPC on Aggregation($p=2\%$)　(b) DPC on Compound($p=2\%$)

(c) DPC on R15($p=2\%$)　(d) DPC on A2($p=2\%$)

图 3 当 $p=2\%$时 DPC 算法在 4 个合成数据集上的聚类结果

Fig. 3 Clustering results of DPC on four datasets when $p=2\%$



(a) NNN-DPC on Spiral　(b) NNN-DPC on Pathbased　(c) NNN-DPC on Aggregation

(d) NNN-DPC on Compound　(e) NNN-DPC on R15　(f) NNN-DPC on A2

图 4 NNN-DPC 算法的聚类结果

Fig. 4 Clustering results of NNN-DPC

在默认参数 $p=2\%$ 的情况下，在 Aggregation，Compound 和 A2 这 3 个数据集上，DPC 算法未给出正确的聚类数和聚类结果，尤其在 Compound 这种同时具有块状、环状结构以及高密度簇被低密度簇包围的复杂数据集上，DPC 算法的一步分配策略不能正确地区分出稀疏和高密度簇，而且高密度簇被错误地分配成了多个簇。

与 DPC 算法相比，NNN-DPC 算法在不指定任何参数的情况下，不仅能够识别正确的聚类数，还能够有效地处理同时具有流形簇、环状簇和块状簇的复杂数据。在 Pathbased 数据集上(见图 4(b))，NNN-DPC 区分出了环形簇和块状簇，并且通过两步分配方法将各个数据点分配给了正确的簇。在 Compound 数据集上(见图 4(d))，NNN-DPC 正确地识别了

图形右上方的稀疏簇和稠密簇，并且将稠密簇中的两个簇准确地合并成了一个簇。

表 2 各算法在合成数据集上的性能指标值

Table 2 Performance indicators values of various algorithms on synthetic datasets

算法	Spiral				Pathbased				Aggregation			
	AMI	ARI	FMI	类数	AMI	ARI	FMI	类数	AMI	ARI	FMI	类数
NNN-DPC	**1.0000**	**1.0000**	**1.0000**	**3/3**	**0.9101**	**0.9292**	**0.9529**	**3/3**	**0.9733**	**0.9788**	**0.9835**	**7/7**
FKNN-DPC	**1.0000**	**1.0000**	**1.0000**	**3/3**	0.7548	0.6834	0.7921	**3/3**	0.9485	0.9686	0.9754	8/7
DPC	**1.0000**	**1.0000**	**1.0000**	**3/3**	0.4997	0.4530	0.6586	**3/3**	0.8060	0.7142	0.7834	9/7
DBSCAN	**1.0000**	**1.0000**	**1.0000**	**3/3**	0.8544	0.8906	0.9281	**3/3**	0.9529	0.9779	0.9827	**7/7**
AP	0.2626	0.1429	0.3202	19/3	0.3525	0.2376	0.4305	16/3	0.5251	0.3473	0.5028	54/7
K-means	−0.0054	−0.0059	0.3277	—	0.5098	0.4613	0.6617	—	−0.7949	0.7300	0.7884	—
算法	Compound				R15				A2			
	AMI	ARI	FMI	类数	AMI	ARI	FMI	类数	AMI	ARI	FMI	类数
NNN-DPC	**0.9921**	**0.9927**	**0.9979**	**6/6**	**0.9938**	**0.9928**	**0.9933**	**15/15**	**0.9693**	**0.9482**	**0.9496**	**35/35**
FKNN-DPC	0.8144	0.8225	0.8734	**6/6**	0.9897	0.9858	0.9868	**15/15**	0.9661	0.9361	0.9379	**35/35**
DPC	0.6322	0.4656	0.5846	9/6	**0.9938**	**0.9928**	**0.9933**	**15/15**	**0.9459**	0.8703	0.8794	32/35
DBSCAN	0.8389	0.8795	**0.9112**	**6/6**	0.9825	0.9819	0.9831	**15/15**	0.6776	0.4000	0.5180	**35/35**
AP	0.4828	0.3024	0.4530	12/6	0.9907	0.9892	0.9899	**15/15**	0.3491	0.3981	0.5011	2391/35
K-means	0.6822	0.5686	0.6669	—	**0.9938**	0.9927	0.9932	—	**0.9781**	**0.9638**	**0.9648**	—

表 2 列出了 NNN-DPC，FKNN-DPC，DPC，DBSCAN，AP，*K*-means 这 6 种算法在 6 个合成数据集上的聚类性能指标，加粗显示的数据为当前数据集中相对最优的指标数据，其中类数指标代表算法最终聚类数与真实类数的比值。对比各算法的 AMI，ARI 和 FMI 这 3 个指标可以发现，即使不输入任何参数，NNN-DPC 算法在各个数据集上都有着更好的聚类效果，相比 KNN-DPC 和 DPC 算法都有很大的提升。

4.2 真实数据集实验

本节从 UCI 数据库[24]中选取了 Ecoli，Wine 等 6 个真实数据集进行实验，以进一步验证 NNN-DPC 算法的性能。这 6 个数据集的数量、维度等具体属性如表 3 所列。

本文使用 NNN-DPC，DPC 等 6 种算法在这 6 个数据集上进行了聚类，沿用 AMI，ARI，FMI 这 3 个聚类指标，并汇总在表 4 中。可以看到，相比 DPC 算法、KNN-DPC 算法，以及 DBSCAN 算法，NNN-DPC 算法在各个数据集上的聚类效果评价指标上都展现出了很大的优势，而且能够更好地识别出实际的聚类数。

由于在现实的聚类任务中无法不断调整参数以取得最佳效果，因此 NNN-DPC 无需参数即可取得良好聚类结果的特性将使该算法在实践中有更加广泛的应用价值。

表 3 UCI 真实数据集

Table 3 UCI real data set

数据集	N	D	Classes	Source
Ecoli	336	7	8	文献[24]
Wine	178	13	3	文献[24]
Dermatology	366	33	6	文献[24]
Breast-cancer	699	9	4	文献[24]
Glass	214	9	7	文献[24]
Parkinson	197	22	2	文献[24]

表 4 各算法在 UCI 真实数据集上的性能指标

Table 4 Performance indicators of each algorithm on UCI real dataset

算法	Ecoli				Breast cancer				Wine			
	AMI	ARI	FMI	类数	AMI	ARI	FMI	类数	AMI	ARI	FMI	类数
NNN-DPC	**0.5866**	**0.7148**	**0.8046**	**6/8**	**0.7295**	0.8203	0.9165	2/4	**0.7579**	**0.7869**	**0.8584**	**3/3**
FKNN-DPC	0.5844	0.6995	0.7971	**6/8**	0.6049	**0.8264**	**0.9173**	6/4	0.6552	0.6735	0.7768	4/3
DPC	0.4892	0.6116	0.7494	3/8	0.0990	0.0933	0.5689	2/40	0.7065	0.6724	0.7835	**3/3**
DBSCAN	0.3772	0.4613	0.6433	3/8	0.6514	0.7128	0.8645	2/4	0.4860	0.4143	0.6480	**3/3**
AP	0.3647	0.2145	0.3799	21/8	0.1410	0.0766	0.2897	76/4	0.3562	0.2689	0.4604	13/3
K-means	0.5181	0.4307	0.5641	—	0.4971	0.7436	0.8746	—	**0.8374**	**0.8592**	**0.9063**	—
算法	Glass				Dermatology				Parkinson			
	AMI	ARI	FMI	类数	AMI	ARI	FMI	类数	AMI	ARI	FMI	类数
NNN-DPC	**0.3306**	0.2011	0.4181	**6/7**	**0.8424**	**0.8497**	**0.8865**	**6/6**	0.1771	0.2686	**0.8140**	**2/2**
FKNN-DPC	0.1933	**0.2085**	**0.5450**	2/7	0.8212	0.7782	0.8306	**6/6**	0.1796	0.3468	0.7879	**2/2**
DPC	0.1987	0.1630	0.4655	3/7	0.5875	0.4905	0.6253	4/6	**0.2295**	0.0869	0.6055	**2/2**
DBSCAN	0.2752	0.2391	0.5248	4/7	0.2321	0.1621	0.4750	5/6	0.0361	−0.0740	0.7124	**2/2**
AP	0.2328	0.1277	0.2916	24/7	0.4957	0.3222	0.4548	22/6	0.0945	0.0306	0.2456	21/2
K-means	0.3048	0.1689	0.3936	—	**0.8734**	0.7294	0.7840	—	0.2129	0.0520	0.5975	—

结束语 针对 DPC 算法对参数敏感以及无法在具有复杂结构的数据集上得到准确聚类结果的缺陷，本文提出了具有无参数特性的 NNN-DPC 算法。实验结果表明，该方法弥补了 DPC 算法这两方面的缺陷，提高了聚类精度，能够在流形数据集、簇间密度差异大的复杂数据集上获得更好的效果。但是，该算法的簇合并准则定义较为简单，对于密度近似且距离较近，甚至部分重合的簇群区分效果不佳，容易误合并这些簇。因此，如何在自然最近邻的基础上，定义更合理、准确的

簇间相似度以及簇合并规则将是下一步研究的工作。

参 考 文 献

[1] MACQUEEN J. Some Methods for Classificati-on and Analysis of MultiVariate Observations[C]// Proc of Berkeley Symposium on Mathematical Statistics & Probability. 1967,1(14):281-297.

[2] KAUFMAN L,ROUSSEEUW P. Clustering by means of medoids[C]// Statistical Data Analysis Based on the L1-norm & Related Methods. North-Holland,1987:405-416.

[3] FREY B J,DUECK D. Clustering by Passing Messages Between Data Points [J]. Science,2007,315(5814):972-976.

[4] ESTER M,KRIEGEL H P,SANDER J,et al. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise[C]// Internation-al Conference on Knowledge Discovery & Data Mining. 1996:226-231.

[5] RODRIGUEZ A,LAIO A. Clustering by fast search and find of density peaks [J]. Science,2014,344(6191):1492-1496.

[6] YANG Z,WANG H J,ZHOU Y. A clusteri-ng algorithm based on cutoff distance and adaptive cluster center [J]. Data Analysis and Knowledge Discovery,2018,2(3):39-48.

[7] WANG Y,ZHANG G Z. Density peak algorithm by automatically determining cluster centers [J]. Computer Engineering and Applications,2018,54(8):137-142.

[8] LIU R,WANG H,YU X. Shared-nearest-neighbor-based clustering by fast search and find of density peaks[J]. Information Sciences,2018,450:200-226.

[9] DU M J,DING S F,JIA H J. Study on density peaks clustering based on k-nearest neighbors and principal component analysis [J]. Knowledge-Based Systems,2016,99:135-145.

[10] XIE J Y,GAO H C,XIE W X,et al. Robust clustering by detecting density peaks and assigning points based on fuzzy weighted *K*-nearest neighbors[J]. Information Sciences,2016,345:19-40.

[11] YANG L J,ZHU Q S,HUANG J L,et al. Adap-tive edited natural neighbor algorithm[J]. Neurocomputing,2017,230:427-433.

[12] HUANG J L,ZHU Q S,YANG L J,et al. QCC:a novel clustering algorithm based on quasicluster centers[J]. Machine Learning,2017,106(3):337-357.

[13] ZHU Q S,FENG J,HUANG J L. Natural neighbor:a self adaptive neighborhood method without parameter *K*[J]. Pattern Recognition Letters,2016,80:30-36.

[14] STEVENS S S. Mathematics,measurement,and psychophysics [C]// Handbook of Experimental Psychology. London:Wiley,1951:1-49.

[15] ZOU X L. Application of Natural Nearest Neighbor in High-dimensional Data Structure Learning[D]. Chongqing:Chongqing University,2011.

[16] HUANG J L. Research on Parameterless Clustering Algorithm Based on Natural Nearest Neighbor[D]. Chongqing:Chongqing University,2014.

[17] 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.

[18] FOWLKES E B,MALLOWS C L. A method for comparing two hierarchical clusterings[J]. Journal of the American Statistical Association,1983,78(383):553-569.

[19] PEDREGOSA F,VAROQUAUX G,GRAMFORT A,et al. Scikit-learn:Machine learning in Python[J]. Journal of Machine Learning Research,2011,12(Oct):2825-2830.

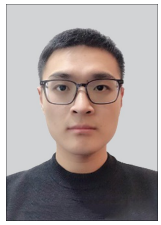
[20] CHANG H,YEUNG D Y. Robust path-based spectral clustering[J]. Pattern Recognition,2008,41(1):191-203.

[21] GIONIS A,MANNILA H,TSAPARAS P. Clusteri-ng aggregation[J]. ACM Transactions on Knowledge Discovery from Data (TKDD),2007,1(1):4.

[22] VEENMAN C J,REINDERS M J T,BACKER E. A maximum variance cluster algorithm[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence,2002,24(9):1273-1280.

[23] FRANTI P,VIRMAJOKI O,HAUTAMAKI V. Fast agglomerative clustering using a k-nearest neighbor graph [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence,2006,28(11):1875-1881.

[24] BACHE K,LICHMAN M. UCI machine learning repository [EB/OL]. [2019-12-20]. http://archive.ics.uci.edu/ml.



TANG Xin-yao, born in 1996, postgraduate. His main research interests include data mining and mechine learning.

ZHANG Zheng-jun, born in 1965, Ph.D, associate professor. His main research interests include data mining and graphic image.