基于主动学习的 K-Hub 聚类算法①

封建邦, 何振峰

(福州大学 数学与计算机科学学院, 福州 350108)

摘 要: K-Hub 聚类算法是一种有效的高维数据聚类算法, 但是它对初始聚类中心的选择非常敏感, 并且对于靠近类边界的实例往往不能正确聚类. 为了解决这些问题, 提出一种结合主动学习和半监督聚类的 K-Hub 聚类算法. 运用主动学习策略学习部分实例的关联限制, 然后利用这些关联限制指导 K-Hub 的聚类过程. 实验结果表明, 基于主动学习的 K-Hub 聚类算法能有效提升 K-Hub 的聚类准确率.

关键词: 高维数据; 半监督聚类; 关联限制; 主动学习; K-Hub

K-Hub Clustering Algorithm Based on Active Learning

FENG Jian-Bang, HE Zhen-Feng

(School of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China)

Abstract: K-Hub is an efficient high-dimensional data clustering algorithm, but it is sensitive to the choice of initial clustering centers and the instances which besides the class border may not be correctly clustered. In order to solve these problems, an improved method which incorporates active learning and semi-supervised clustering into K-Hub clustering algorithm is proposed. It uses active learning strategy to study pairwise constraints, and then, it uses these pairwise constraints to guide the clustering process of K-Hub. The experiment results demonstrate that the improved method can enhance the performance of K-Hub clustering algorithm.

Key words: high dimensional data; semi-supervised clustering; pairwise constraints; active learning;K-Hub

1 引言

随着科技的发展, 越来越多的实际应用需要处理维度很高的数据, 如人脸识别、生物信息数据识别等需要处理成百上千维的高维数据. 在数据挖掘、机器学习领域中, 高维数据分析会遇到低维数据分析不会遇到的问题, 即随着维度的增加, 数据空间急剧增大, 导致数据变的稀疏, 出现了许多空空间的现象, 这些现象称为“维度灾难”[1]. 高维数据的“维度灾难”现象对很多机器学习任务提出了严峻的挑战, 如最近邻搜索、离群点检测、贝叶斯建模等. Hubness 现象是与最近邻有关的维度灾难的一方面[2]. 用 NK(x)表示数据集 S 中某一个实例 x 出现在其他实例的 K 近邻列表中的次数. Hubness 现象是指在高维数据空间中 NK(x)的分布呈现出明显的右偏, 而且这种右偏程度与数据内在维度呈正相关关系, 进而导致少量实例频繁的出现在其他实例的 K 近邻列表中. 2009 年至 2014 年, Radovanovic, Ivanovic 等探索了 Hubness 现象的起因和影响[3,4,5], 发现 Hubness 是高维数据的一种内在属性, 并且对机器学习算法产生了许多影响. 基于这种情况, 目前高维数据 Hubness 现象已经成为一个研究热点, 并提出了很多相关的分类、聚类、实例选择等算法[4-6].

传统的聚类算法是一种无监督的学习方法, 它把相似的实例划分到同一类中, 不相似的实例划分到不同类中. 然而, 随着对聚类研究的深入, 研究者认为聚类实际上应该是主观的, 对同一个数据集, 不同的应用, 其相应的聚类结果应该不同, 如对于鲸鱼、大象、金枪鱼等, 如果按照是否为哺乳动物进行聚类, 则鲸鱼和大象应该聚为一类; 而根据是否生活在水中为标准, 则鲸鱼和金枪鱼应该聚为一类[7]. 为了在聚类分析时,有效结合用户倾向, 半监督聚类算法被提出.

① 收稿时间:2015-07-05;收到修改稿时间:2015-09-08

半监督聚类算法是在传统的无监督聚类算法的基础上, 结合了一些关于聚类的有限的背景知识, 也就是用户倾向. Wagstaff 引入了数据之间的关联限制(成对约束)表示聚类的背景知识: Must-Link(正关联)和 Cannot-Link(负关联)表示两个实例之间的关系[8], 有 Must-Link 限制的两个实例一定属于同一类, 有 Cannot-Link 限制的两个实例一定属于不同类, 这两种限制都具有对称性和传递性.

在半监督聚类中, 需要在聚类时引入关联限制, 由于获取大量的关联限制的成本是非常高的, 并且不同的关联限制对聚类的提升效果不一样, 因此, 为了获取对聚类效果有明显提升的关联限制, 国内外开始研究主动学习在半监督聚类中的应用.

到目前为止, 主动学习在聚类中应用可以分为两类. 第一类是在聚类执行之前, 先依据一定的规则从数据集中选出一部分实例, 由专家给出它们的关联限制, 然后用这些关联限制进行半监督聚类. Basu 等人在文献[9]中提出的一种结合主动学习的半监督聚类算法框架, 其主动学习策略是一种两阶段的基于最远距离优先的主动学习策略; 第二类是在聚类的每一次迭代过程中, 根据当前聚类结果选出能提升聚类效果的实例, 由专家给出它们的关联限制. Huang 等人在文献[10]中, 详细讨论了根据当前聚类结果, 采取不同的主动学习策略(不确定性取样、最小泛化误差)对聚类准确率的影响.

目前, 已经有很多对高维数据的 Hubness 特性、半监督聚类、主动学习的研究, 但是它们并没有把高维数据的 Hubness 特性与具体的半监督聚类、主动学习算法相结合. 因此, 我们研究在基于高维数据 Hubness 特性的聚类算法 K-Hub 中引入半监督聚类以及主动学习的思想, 对 K -Hub 聚类算法的聚类准确率的影响.

2 K-Hub聚类算法及不足

2.1 K-Hub 聚类算法

K-Hub 聚类算法是一种基于 Hubness 的高维数据聚类算法, 它与经典的 K-Means 聚类算法很相似, 不同之处在于它们的聚类中心点的更新方式不同. K-Means 聚类算法使用当前簇中所有实例的平均值作为下一次的聚类中心点, 而 K-Hub 聚类算法则选择当前簇中 Hubness 得分最高的实例作为下一次聚类的中心点. 由于在 K-Hub 聚类算法中, 需要用常数表示类数和近邻数, 为了进行区别, 在下文中, 用K_0表示类数, 用 K 表示近邻数. K-Hub 聚类算法的主要步骤如下:

输入: 数据集 S, 类数 K_0, 最大的迭代次数 MAXITER;

输出: 一个数据划分 P;

1) 随机选择 K_0 个实例作为初始聚类中心;

2) 把实例划分到离聚类中心点最近的簇中;

3) 选出每个簇中 Hubness 得分最高的实例, 作为下一次聚类的中心点;

4) 如果聚类中心点不变或者达到最大迭代次数, 输出当前划分, 否则执行步骤 2);

2.2 Hubness 得分

K-Hub 聚类算法认为一个实例出现在其他实例的 K 近邻列表中的次数越多($N_K(x)$值越大), 实例的 Hubness 得分越高. 论文[11]讨论了这种计算方法存在的一些弊端, 并且引入轮廓系数来计算实例的 Hubness 得分, 提出了两种不同计算实例 Hubness 得分的方法: 轮廓系数方法(silhouette index approach)、带权重的相对 Hubness 值方法(weighted relative hubness approach).

1) 轮廓系数方法

计算每个实例的 $N_K(x)$值, 如果在同一个簇中, 有多个实例的 $N_K(x)$值相同, 选择其中轮廓细数较大的实例作为聚类的中心点. 轮廓系数是一种聚类质量评判标准, 已经被广泛应用于高维数据分析中[3,5], 对于一个特定的实例 x, 其轮廓系数的计算方法:

$$SI(x)=\frac{b(x)-a(x)}{\max\{a(x),b(x)\}} \tag{1}$$

如果用实例到簇中所有实例距离的平均值表示实例到某一个簇的距离, 那么 $a(x)$表示实例 x 到它所在簇的距离, $b(x)$表示实例 x 到其他簇的最小距离. 如果实例 x 是聚类的中心点, 那么我们期望实例 x 与它所在簇的实例比较相似的同时与其他簇的实例有较大差异, 即 $a(x)$较小而 $b(x)$较大. 因此, 在实例的 $N_K(x)$值相同的情况下, $SI(x)$值更大的实例更有希望是类中心点.

2) 带权重的相对 Hubness 值方法

对于给定的数据集 S, $x,y\in S$, 实例 x 的 Hubness 得分的计算方式如下:

$$f(x)=\sum_{y\in S}\delta(x) \tag{2}$$

$$\delta(x)=\begin{cases}SI(x): x\in NN_K(y)\text{近邻}\\ 0: x\notin NN_K(y)\end{cases} \tag{3}$$

其中 $SI(x)$表示实例 x 的轮廓系数, $NN_K(y)$表示实例 y 的 K 近邻.

论文[11]用这两种计算方法在模拟数据和实际数据中进行了实验对比, 发现采用带权重的相对 Hubness 值的计算方法取得了更好的聚类准确率, 因此本文也采取这种方法计算实例的 Hubness 得分.

2.3 K-Hub 聚类算法的不足

K-Hub 聚类算法模型简单, 主要缺点有两点:

1) 聚类效果对初始聚类中心非常敏感[5,12]. 如果有多个初始聚类中心属于同一类, 那么在后续的迭代过程中, 其他类中可能没有实例被选为聚类中心, 那么算法的聚类效果将会很差. 如图 1 所示, 数据集共有三类, 实心圆表示初始聚类中心, 空心圆表示数据集中的其他实例, 从图中可以看出, 初始聚类中心全部在左上方的类中, 并且该类的实例明显比其他两个类密集, 导致该类中实例的 Hubs 得分明显高于其他两类中的实例. 当出现这种情况时, 聚类的中心可能会一直在左上方的类中产生, 以致其他两个类没有实例成为聚类中心点, 从而影响聚类准确率.

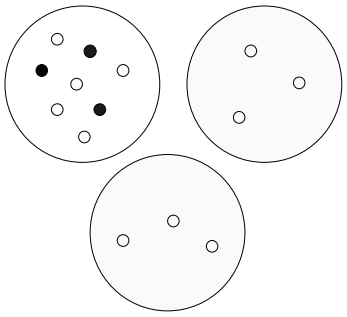

图 1 初始聚类中心对 K-Hub 聚类算法的影响

2) 对靠近类边界的实例, 聚类效果较差. 对于靠近类边界的实例容易出现错误划分的情况, 这样会影响到聚类的准确率. 如图 2 所示, 数据集有两类, 空心实例表示靠近类边界的实例, 实线表示实际的类划分, 虚线表示 K-Hub 算法得出的划分, 从图中可以看出, K-Hub 算法得出的划分对靠近类边界的实例出现了错误划分的情况.

3 基于主动学习的K-Hub聚类算法

为了解决 K-Hub 聚类算法存在的问题, 我们引入主动学习策略对 K-Hub 算法进行改进, 应用主动学习从数据集中选取少部分实例, 由专家给出它们的关联限制, 并利用这些关联限制指导 K-Hub 算法的执行过程, 进行半监督聚类. 那么我们需要解决的问题有两个: 1)如何选取实例向专家询问; 2)如何把得到的关联限制应用到 K-Hub 算法中去. 目前已经有不少算法把主动学习结合到聚类算法中去[9,10,13,14]. Basu 在文献[9]中提出一种结合主动学习的半监督聚类算法框架(Active Semi-Supervision for Pairwise Constrained Clustering, 我们简称为 ASPCC), 可以很好的解决这两个问题. 然而, Basu 提出的方法并没有考虑高维数据的 Hubness 特性, 因此为了提升 ASPCC 算法框架在 K-Hub 聚类算法中的表现, 我们结合高维数据的 Hubness 特性对其进行改进, 并将改进后的算法命名为 HASPCC. 首先我们先介绍一下 Basu 提出的算法 ASPCC, 再给出我们改进后方法 HASPCC, 最后再给出基于主动学习的 K-Hub 聚类算法的详细步骤.

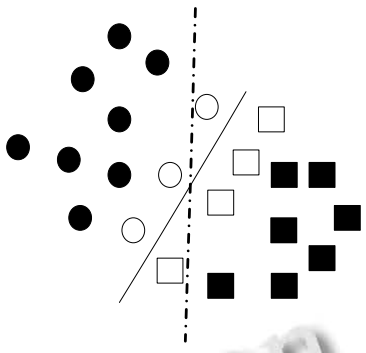

图 2 K-Hub 算法对靠近边界实例的划分

3.1 ASPCC 算法

3.1.1 主动学习策略

在选择实例方面, Basu 采用的是一种两阶段的基于最远距离优先的主动学习策略. 在第一个阶段中, 通过一个称为“Explore”的步骤, 得到 K_0 个两两不相交的等价类, 每个等价类都属于一个不同的类; 第二个阶段使用“Consoliadte”对这 K_0 个等价类进行扩充, 扩展关联限制集合, 以更好的发现数据集的类分布.

第一阶段(Explore):

输入: 数据集 S, 类数 K_0, 关联限制数量 Q;

输出: $\lambda(\lambda \le K_0)$个不同的等价类 $T\{T_1, T_2, \ldots T_\lambda\}$, 每个等价类至少包含一个实例;

1) 初始化 K_0 个等价类 T_i, $(i=1,2\ldots K_0)$为空

2) 从数据集 S 中随机选一个实例加入 T_1,λ=1,用 q 表示已获取的关联限制数量, 并初始化 q=0;

3) while $q<Q$ and $\lambda<K_0$

从数据集 S 中其他实例中选择距离 T 中所有等价类最远的实例 x

if x 与所有等价类都是 Cannot-link 关系

新建一个等价类,$\lambda\leftarrow\lambda+1$;

else

把 x 加入到与其存在 Must-link 关系的等价类中;

$q\leftarrow q+1$;

第二阶段:

输入: 数据集 S, 类数 K_0, 关联限制数量 Q, K_0 个等价类 $T\{T_1, T_2,...T_{K0}\}$;

输出: K_0 个等价类, 每个等价类比输入的等价类包含更多实例

1)计算每个等价类的中心点 C_i;

2)如果获取的关联限制数量小于 Q, 随机选择一个实例 x, 计算其与 C_i 的距离, 并根据距离从小到大判断实例 x 是否与等价类 T_i 存在 Must-link 关系, 如果存在, 则实例 x 加入 T_i;

3.1.2 半监督策略

在使用关联限制进行半监督聚类方面, Basu 通过两个步骤来实现半监督聚类: 第一、利用关联限制初始化聚类中心, 以保证初始聚类中心属于不同的类; 第二、在聚类算法原有的目标函数基础上加上惩罚项(即当实例的聚类结果与已知的关联限制冲突时, 需要付出的代价), 以更好的满足已知的关联限制. 算法中假设 M 表示 Must-link 集合, $(x_i,x_j)\in M$ 表示 x_i,x_j 应该要划分到同一个类中; 用 C 表示 Cannot-link 集合, $(x_i,x_j)\in C$ 表示 x_i,x_j 应该划分到不同类中. 用 $w_{ij}v[l_i\neq l_j]$ 表示存在 Must-link 关系的两个实例被划分到不同类的代价, $w_{ij}^{'}v[l_i=l_j]$ 表示存在 Cannot-link 关系的两个实例被划分到同一个类中的代价. 其中 v 是一个指示函数, $v[true]=1$, $v[false]=0$. PCC 就是通过最小化目标函数 $f(x)$ 来指导聚类过程的, 其中 x_i 被划分到类 u_j 中:

$$f(x)=\frac{1}{2}\sum_{x_i\in X}\|x_i-u_j\|^2+\sum_{(x_i,x_j)\in M}w_{ij}v[l_i\neq l_j]+\sum_{(x_i,x_j)\in C}w_{ij}^{'}v[l_i=l_j] \quad (4)$$

通常取 $w_{ij}=w_{ij}^{'}=w$.

3.2 改进的 ASPCC 算法 HASPCC

ASPCC 算法虽然能搞提高聚类效率, 但是仍然存在一些问题. 第一、在主动学习过程中并没有考虑高维数据的 Hubness 特性以及聚类的具体情况. 根据已有的研究结果, 主动学习核心是从数据集中选出价值最大实例, 并加以利用, 以提高算法的整体表现. 在高维数据中, Hubs 实例更靠近类中心, 而 K-Hub 聚类算法也是选择 Hubs 实例作为类中心聚类, 故而, 相比其他实例, Hubs 实例的价值更大; 另外, 在聚类的每次迭代结束时, 存在一部分实例的聚类结果具有较高的不确定性, 因此, 如果能得到这些不确定性较高的实例的关联限制, 就能进一步提高最终的聚类. 基于以上原因, 我们需要在主动学习过程中考虑高维数据的 Hubness 特性以及聚类的具体情况. 第二、在应用已获取的关联限制进行半监督方面, ASPCC 算法在计算实例的聚类结果与已知关联限制发生冲突的代价时, 并没有区分数据集中的 Hubs 实例与非 Hubs 实例. 由于 Hubs 实例更靠近类中心, 我们认为, Hubs 实例被错误划分对聚类结果带来的不良影响应该要比非 Hubs 实例大. 因此, 在应用关联限制方面, 我们也需要考虑高维数据的 Hubness 特性. 然而, 传统的 ASPCC 算法并没有考虑这些因素, 为此, 我们对 ASPCC 算法进行改进, 在主动学习策略学习实例间的关联限制以及应用这些关联限制进行半监督聚类两方面, 结合高维数据的 Hubness 特性以及聚类的具体情况进行改进, 使其更适合 K-Hub 聚类算法.

3.2.1 改进的主动学习策略

在选择实例由专家给出关联限制方面, PCC 算法分两个阶段, 第一个阶段获得 K_0 个两两不相交的等价类, 每个等价类都属于不同的类; 第二个阶段对这 K_0 个等价类进行扩充, 扩展关联限制集合, 我们对这两个阶段都进行相应的改进.

首先, 在第一个阶段(Explore), 结合高维数据 Hubs 实例更靠近类中心的特点, 不再从整个数据集选择实例, 而是把选择范围缩小的 Hubs 实例中. 为此将“Explore”中的步骤 1)修改为:

1) 初始化每个 T_i 个等价类为空;

2) 计算数据集 S 中每个实例的 $N_K(x)$ 值, 并将 $N_K(x)$ 大于某一个阀值 N 的实例集合作为数据集的 Hubs 实例集合 H(在实验中, 取 $N=K$);

然后将“Explore”的步骤 2)和步骤 3)中的实例选择

范围从整个数据集 S 改为数据集的 Hubs 实例集合 H, 我们把改进过的这个步骤称为基于 Hub 的“Explore”, 简称为“HExplore”.

其次, 在第二阶段(Consolidate), PCC 算法在扩充关联限制集合时, 并没有结合聚类的具体情况, 这样不利于靠近类边界实例的聚类. 因此, 我们把第二个阶段结合到聚类的迭代过程中去, 在聚类的每一次迭代结束时, 根据当前聚类结果, 使用公式(1)计算所有实例的轮廓系数, 并且取其倒数作为实例的不确定性, 然后选择不确定性最高的实例对现有关联限制进行扩展. 因为轮廓系数越小的实例, 当前聚类越不能确定其所属类, 其不确定性越高. 该步骤改进后, 将其命名基于不确定性的“Consolidate”, 简称为“UConsolidate”具体步骤如下:

输入: 数据集 S, 类数 K_0, 关联限制数量 Q, K_0 个等价类 $T\{T_1, T_2,...T_{K0}\}$, 已获取的关联限制数量 q;

输出: K_0 个等价类, 每个等价类比输入的等价类包含更多实例

if $q < Q$

根据当前聚类结果计算每个实例的不确定性;

选择最高的实例 x, 向专家询问, 判断 x 是否与等价类 T_i 存在 Must-link 关系, 如果存在, 则实例 x 加入 T_i;

$q \leftarrow q+1$;

3.2.2 改进的半监督策略

在应用已获取的关联限制进行半监督聚类方面, ASPCC 中将聚类的目标函数修改为公式(4), 使聚类结果更符合关联限制集合. 在 ASPCC 中, w 的取值是固定的(在我们的实验中, 取 w 为数据集中所有实例间距离的平均值), 即对于数据集中的所有实例, 其聚类结果与关联限制发生冲突的代价是一样的, 然而实际情况并非如此. 考虑到在高维数据中, Hubs 实例更靠近类中心, 因此, 相对于其他实例, 当 Hubs 实例被划分到错误的类中, 可能会产生更严重的后果. 假设, 按照已有的关联限制, Hubs 实例 x 应该属于类 C_1, 但是算法执行时, 却把实例 x 划分到类 C_0 中, 那么再下一次迭代时, 实例 x 有可能被选为类 C_0 的聚类中心点, 导致聚类中心没有属于 C_0 的实例, 那么对于 C_0 的聚类效果将会非常差. 如果要避免这种情况, 在对实例进行划分时, Hubs 实例的划分违反已知的关联限制时, 就必须付出更大的代价. 为了解决这个问题, 我们将 ASPCC 的目标函数中的 w 设置成能够区分高维数据中 Hubs 实例的 w_x, 并且用公式(5)计算 w_x 的值.

$$w_x = w \times \frac{N_K(x)}{K} \tag{5}$$

其中 w 的取值方式与 ASPCC 相同, $N_K(x)$表示实例 x 出现在其他实例 K 近邻列表中的次数. 由于高维数据中 Hubs 实例的 $N_K(x)$相比其他实例更大, 相应的 w_x 值也更大, 因此 Hubs 实例违反关联限制需要的代价就更大.

3.3 改进后的 K-Hub 聚类算法

主动学习在 K-Hub 聚类算法中的应用是先通过“HExplore”从数据集中学习一部分 Hubs 实例之间的关联限制, 并利用这些关联限制选择 K-Hub 聚类算法的初始聚类中心, 然后在聚类的迭代结束时, 通过“UConsolidate”对现有关联限制集合进行扩展, 算法的具体步骤如下:

输入: 数据集 S, 类数 K_0, 最大迭代次数 $MAXITER$, 关联限制集合大小 Q

输出: 一个数据划分 P;

1) 使用 HExplore 获取一部分 Hubs 实例的关联限制;

2) 根据已有的关联限制集合, 计算出等价关系集合 $\{T_1,T_2,...T_\lambda\}$;

3) if $\lambda \geq K_0$

分别取 K 个等价类中 $N_{K(x)}$最大的实例作为初始聚类中心;

else if $\lambda < K_0$

分别取 λ 个等价类中 $N_{K(x)}$最大的实例作前 λ 个初始聚类中心;

if 存在实例 x 与所有等价类是 Cannot-link 关系则把 x 初始化为第 λ+1 个类中心;

4) 从数据集中随机选择实例初始化其他的类中心对于每个实例 x, 划分到类 l*中, 其中

$$l^* = \arg\min_l \left(\frac{1}{2}\| x-u_l \|^2 + w_x \sum_{(x,x_j)\in M} v[l \neq l_j] + w_x \sum_{(x,x_j)\in C} v[l = l_j]\right)$$

5) 如果已获取关联限制数量小于 Q, 使用 UConsolidate 扩展关联限制集合

6) 选出每个簇中 Hubness 得分最高的实例, 作为下一次聚类的中心点;

7) 如果聚类中心点不变或者达到最大迭代次数,

输出当前划分, 否则, 执行 4);

4 实验

本文采用 8 个 UCR 时间序列数据集进行实验, 用兰德系数(Rand Index)来计算聚类的准确率. 对于一个给定的数据划分, 用 a 表示属于同一类也被划分到同一个簇中的实例对个数; 用 b 表示不属于同一类也被划分到不同簇的实例对的个数; 用 c 表示属于同一类但被划分到不同簇的实例对的个数, 用 d 表示不属于同一类但被划分到同一个簇的实例对的个数; 最后用 $(a + b)/(a + b + c + d)$计算兰德系数, 所有实验都采用 10 次 10-折交叉验证计算结果. 表 1 给出了实验所用的数据集的信息.

表 1 实验所采用的数据集

数据集	类数	大小	简称
CBF	3	930	CBF
ECGFiveDays	2	884	ECG
ItalyPowerDemand	2	1096	Italy
MALLAT	8	1024	MAL
CinC_ECG_torso	4	1420	CinC
Symbols	6	1020	Sym
SonyAIBORobotSurfaceII	2	980	Sony
TwoLeadECG	2	1162	Two

在实验中, 我们采用主动学习策略学习数据集中 10%的实例的关联限制, 近邻数 K 设置为 5, 每次聚类算法都在迭代次数达到 50 次或者聚类结果不再发生变化时结束.

在实验中, 我们比较随机选择实例获取关联限制(random)、采用 Basu 提出的 ASPCC 算法以及我们改进过的 ASPCC 算法(HASPCC)对初始 K-Hub 聚类算法进行改进后的聚类结果, 实验结果如表 2 所示, HASPCC 算法对 K-Hub 的聚类准确率提升最高.

表 2 不同实例选择策略对算法准确率的影响

数据集	K-Hub	Random	ASPCC	HASPCC
CBF	62.43	63.52	64.73	**66.34**
ECG	49.97	51.12	52.39	**53.62**
Italy	74.38	76.29	76.73	**77.75**
MAL	81.23	82.84	83.65	**85.20**
CinC	83.43	85.26	86.19	**87.49**
Sym	68.58	70.82	72.21	**73.38**
Sony	65.24	66.35	67.37	**68.76**
Two	59.31	61.45	62.89	**63.93**

5 结语

K-Hub 聚类算法的主要问题就是它对初始聚类中心点的选择非常敏感, 并且对靠近类边界的实例聚类效果较差. 本文针对上述问题, 引入主动学习策略对 K-Hub 聚类算法进行改进, 并且在主动学习过程中结合高维数据的 Hubness 特性以及当前的聚类结果, 选择靠近类中心的 Hubs 实例和靠近类边界的实例, 由专家给出它们之间的关联限制, 并利用这些关联限制, 采用半监督聚类的思想提高 K-Hub 聚类算法的准确率. 实验表明, 基于主动学习的 K-Hub 聚类算法的聚类准确率有了显著提高.

参考文献

1 Donoho DL. High-dimensional data analysis:The curses and blessings of dimensionality. AMS Math Challenges Lecture, 2000: 1–32.

2 Radovanovic M, Nanopoulos A, Ivanovic M. Nearest neighbour in high-dimensional data: The emergence and influence of hubs. Proc. of 26th Annual International Conference on Machine Learning(ICML), 2009:865–872.

3 Radovanovic M, Nanopoulos A, Ivanovic M. Hubs in space: Popular nearest neighbors in high-dimensional data. Journal of Machine Learning Research, 2010, 11: 2487–2531.

4 Tomasev N, Radovanovic M, Mladenic D, Ivanovic M. Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification. International Journal of Machine Learning and Cybernetics, 2014, 5(3): 445–458.

5 Tomasev N, Radovanovic M, Mladenic D, Ivanovic M. The role of hubness in clustering high-dimensional. Knowledge and Data Mining, 2014, 26(3): 739–751.

6 Zhai TT, He ZF. Instance selection for time series classification based on immune binary particle swarm optimization. Knowledge-Based Systems, 2013, 49: 106–115.

7 何振峰,熊范纶.结合限制的分隔模型及 K-Means 算法.软件学报,2005,16(5):799–809.

8 Wagstaff K, Cardie C. Clustering with instance-level constraints. Proc. of the Seventeenth International Conference on Machine Learning(ICML 2000), 2000. 1103–1110.

9 Basu S, Banerjee A, Mooney J. Active semi-supervision for pairwise constrained clustering. Proc. of the Society for Industrial and Applied Mathematics(SIAM) Int Conf, on

Data Mining. 2004. 333–344.

10 Huang RZ, Lam W, Zhang Z. Active learning of constraints for semi-supervised text clustering. Proc. of the Society for Industrial and Applied Mathematics(SIAM) Int'1 Conf. on Data Mining. 2007. 113–124.

11 He ZF. Hub Selection for hub based clustering algorithms. Proc. of International Conference on Fuzzy System and Knowledge Discovery(FSKD). 2014. 479–484.

12 张巧达,何振峰.基于 Hub 的高维数据初始聚类中心的选择策略.计算机系统应用,2015,24(4):171–175.

13 Huang R, Lam W. Semi-supervised document clustering via active learning with pairwise constraints. Proc. of International Conference on Date Mining(ICDM), 2007: 517–522.

14 赵卫中,马慧芳,李志清,史忠植.一种结合主动学习的半监督文档聚类算法.软件学报,2012,23(6):1486–1499.