Computer Science and Application 计算机科学与应用, 2020, 10(5), 906-913
Published Online May 2020 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2020.105093

Hans汉斯

Application and Research of DBSCAN Optimization Algorithm in Big Data Analysis of Experimental Text

Tingting Shi[1*], Weihua Liu[2#], Shuangyin Liu[1,3,4], Longqin Xu[1]

[1]College of Information Science and Technology, Zhongkai University of Agriculture and Engineering, Guangzhou Guangdong
[2]Guangdong Vocational College of Judicial Police, Guangzhou Guangdong
[3]Guangdong Provincial Agricultural Products Safety Big Data Engineering Technology Research Center, Zhongkai University of Agriculture and Engineering, Guangzhou Guangdong
[4]Smart Agriculture Engineering Technology Research Center of Guangdong Higher Education Institues, Zhongkai University of Agriculture and Engineering, Guangzhou Guangdong
Email: #lwhhao@126.com

Received: Apr. 20th, 2020; accepted: May 5th, 2020; published: May 12th, 2020

Abstract

Big data is a research hotspot emerging in the computer field in recent years. Clustering can solve problems in the field of big data, such as data mining, machine learning, and text processing. Aiming at the problems that parameters of traditional DBSCAN algorithm need to be set manually and the algorithm speed cannot adapt to the application of big data, a DBSCAN optimization algorithm was proposed. The KD tree was used to speed up the search for neighborhood objects, significantly reducing the running time of the algorithm; at the same time, the density threshold (Minpts) was adaptive by calculating the mathematical expectations of all neighborhood objects; then, a text clustering process was designed, and the weights of feature items were optimized through SD-TF-IDF to complete the text clustering task; finally, it was applied to the mining and analysis of big data of computer experimental text in colleges and universities, and good results had been achieved.

Keywords

DBSCAN, Density Clustering, Text Clustering, Experimental Big Data Analysis

*第一作者。
#通讯作者。

文章引用: 史婷婷, 刘卫华, 刘双印, 徐龙琴. DBSCAN 优化算法在实验文本大数据分析中的应用研究[J]. 计算机科学与应用, 2020, 10(5): 906-913. DOI: 10.12677/csa.2020.105093

DBSCAN优化算法在实验文本大数据分析中的应用研究

史婷婷[1*]，刘卫华[2#]，刘双印[1,3,4]，徐龙琴[1]

[1]仲恺农业工程学院信息科学与技术学院，广东 广州
[2]广东司法警官职业学院，广东 广州
[3]仲恺农业工程学院广东省农产品安全大数据工程技术研究中心，广东 广州
[4]仲恺农业工程学院广东省高校智慧农业工程技术研究中心，广东 广州
Email: #lwhhao@126.com

收稿日期：2020年4月20日；录用日期：2020年5月5日；发布日期：2020年5月12日

摘 要

大数据是近年来计算机领域兴起的研究热点，通过聚类可以解决诸如数据挖掘、机器学习、文本处理等大数据领域问题。针对传统的DBSCAN算法参数需要人工设定，且算法速度无法适应大数据应用等问题，本文提出了一种DBSCAN优化算法。利用KD树加快查找邻域对象，显著减少算法的运行时间；同时，通过计算所有邻域对象的数学期望，实现密度阈值(Minpts)参数自适应；接着，设计了一种文本聚类流程，通过SD-TF-IDF算法对特征项的权值进行优化，进而完成对文本的聚类任务；最后，将其应用于高校计算机实验文本大数据的挖掘分析中，取得了良好的效果。

关键词

DBSCAN、密度聚类、文本聚类、实验大数据分析

1. 引言

在云计算、物联网等技术的带动下，社会已经步入大数据时代。计算机实验在其组织、运行、实施过程中会产生丰富的文本数据，如学生遇到的理论知识点问题、实践操作问题、实验进度、实验时间、实验结果和创新性想法等，且实验中的问题往往依靠师生口头交流解决，缺乏相应数据记录，更无法进行数据分析。如何从产生的文本大数据中提取出隐含的、有价值的信息，提升学生对知识点的理解、提高学生的学习效率和教师的教学质量，是目前高校计算机实验教学亟待解决的问题。

为了解决上述问题，本文提出了一种 DBSCAN 优化算法；同时，设计了一种文本聚类流程。首先，将文本进行分词处理以提取特征；接着，利用 SVM 将文档映射为一个特征向量，并且通过 SD-TF-IDF 算法对特征项的权值进行优化；利用 DBSCAN 优化算法对文本进行聚类，计算文本相似度，将相似度低于阈值的文本作为孤立点进行处理[1]，来寻找核心数据对象的密度可以达到的数据点过程，从而实现对

高校计算机实验文本大数据的智能化管理。

2. DBSCAN 聚类算法

DBSCAN 全称为 Density-Based Spatial Clustering of Applications with Noise，这是一类基于密度的聚类算法，由 Martin Ester、Hans-PeterKriegel [2]等人在 1996 年提出。其与划分和层次聚类方法不同，不需要预先指定簇的个数，将簇定义为密度相连的点的最大集合，能够把具有足够高密度的区域划分为簇，并可在噪声的空间数据库中发现任意形状的聚类[3]，通过过滤低密度的样本区域，来发现稠密的样本区域。

该算法是基于一组“邻域”参数(Eps, Minpts)来描述样本分布的紧密度，假设给定样本集合 $D=\{p_1,p_2,\cdots,p_n\}$，DBSCAN 聚类算法涉及的相关定义如下：

定义 1 Eps-邻域：空间中任意给定对象半径 Eps 内的邻域称为该对象的 Eps-邻域；

定义 2 核心对象：如果给定对象 Eps-邻域内的样本点数大于等于密度阈值 Minpts，则称该对象为核心对象；

定义 3 直接密度可达：对于样本集合 D，如果样本点 q 在 p 的 Eps-邻域内，并且 p 为核心对象，那么对象 q 从对象 p 直接密度可达；

定义 4 密度可达：对于样本集合 D，给定一串样本点 $p_1,p_2,\cdots,p_m$，$p=p_1$，$q=p_m$，假如对象 pi 从 pi-1 直接密度可达，那么对象 q 从对象 p 密度可达；

定义 5 密度相连：存在样本集合 D 中的一点 o，如果对象 o 到对象 p 和对象 q 都是密度可达的，那么 p 和 q 密度相连；

定义 6 边界点：若对象 p 不是核心点，对象 p 密度可达其他核心对象，则将对象 p 称为边界点。

定义 7 噪声点：既不是核心点也不是边界点的点为噪声点。

DBSCAN 聚类算法的核心思想是：先发现密度较高的点，然后把相近的高密度点逐步连成一片，进而生成各种簇。

3. DBSCAN 聚类算法的优化

3.1. DBSCAN 优化算法的思想

根据聚类假设原理，同一类的文本相似度比较大，不是同一类的文本之间的相似度就比较小[4]。聚类算法是一种自主的无监督机器学习方式，聚类不用提前对文本进行手工标注类别，也无需经过训练样本。因此，它具有比较高的自动化处理能力和灵活性，是现在对文本信息进行有效组织、导航和摘要的重要手段之一[5]。

然而，传统的 DBSCAN 算法需要在无先验知识情况下人工设定 Eps 和 Minpts 参数，如果参数选择不合理将导致聚类结果不理想；同时，DBSCAN 算法虽然聚类速度快，且能有效处理噪声点以及发现任意形状的空间聚类，但当数据集的数据量比较大的时候，查找邻域数据对象所消耗的时间会变得越来越多，降低算法运行效率。

因此，本文提出了一种 DBSCAN 优化算法，较好地解决了算法参数自适应和算法速度问题。首先，将数据对象放入一颗 KD 树中，加快查找到邻域对象，使得算法的运行时间大大减少；同时，在查找邻域数据对象的过程中，将数据对象之间的欧几里得距离与 Eps 参数做比较，以确定其是否为当前数据对象的邻域对象。在每次比较中，计算所有邻域对象的数学期望，即可实现 Minpts 参数自适应。

3.2. DBSCAN 优化算法的步骤

根据上述算法思想，DBSCAN 优化算法的具体步骤如下：

输入：含有 n 个数据对象的数据集 D，半径参数 Eps。

输出：生成满足密度要求的簇。

步骤 1：将数据集合 D 中 n 个数据对象构建成一棵 KD 树；

步骤 2：遍历数据集合 D，从 KD 树中搜索当前数据对象 p 的最近邻域对象也就是对象 p 的密度可达对象，得到当前数据对象 p 的邻域；

步骤 3：统计每个数据对象 p 的邻域数据对象数量，得到数量 i；

步骤 4：计算邻域数据对象的数学期望，得到 Minpts=i/n，返回 Minpts；

步骤 5：从数据集合 D 中随机抽取一个为被处理的对象 p，且在它的 Eps-近邻满足密度阈值要求称为核心点；

步骤 6：遍历数据集合 D，从 KD 树中搜索当前数据对象 p 的最近邻域对象也就是对象 p 的密度可达对象，形成一个新的簇；

步骤 7：通过密度相连产生最终簇结果；

步骤 8：重复执行步骤 3 和步骤 4，直到数据集中 D 中所有数据对象都标记为“已处理”。

4. 基于 DBSCAN 优化算法的文本聚类

为了测试本文算法性能，设计了一个基于 DBSCAN 优化算法的文本聚类流程。

4.1. 文本预处理

对文本数据进行聚类前，需要对其进行预处理。

4.1.1. 文本特征提取

首先，对文本进行分词处理。如中文文档就通过使用砌词和字典等方式进行分词操作，计算机会自动运行分词的过程[6]。因为文档数据中会存在没有太大意思的符号和词汇等[7]，所以在分词的过程中，应去除掉这些没有意义的符号和词汇；同时，文档中还有存在多个同义词的情况，因此需要将这些同义词合并操作，如“信息”与“讯息”就是属于同义词，所以可将它们合并成“信息”[8]。

经过分词操作后，出现一大推的特征词汇，如词组或者词条等，大部分的特征词汇对文本的挖掘没有太大的意义，就需要对这些词汇进行按需的筛选，这样才能有效地提高聚类算法的性能和精确性[9]；接着，本文使用 TF-IDF 算法找出文本中的关键词，将权值最大的 n 个关键词作为特征项代表文本进行聚类。

4.1.2. 文本特征向量化

本文使用空间向量模型 VSM (Vector Space Model)将文档转化成 n 维特征向量空间。VSM 把每篇文档都表示为特征项-权值向量，把文本看作是一系列特征项 t 的集合，对每个特征项赋予对应的权值[10]。

文档 x 映射为 n 维向量 $\vec{x}$：

$$\vec{x}=\left(\left(t_1,w_{x1}\right),\left(t_2,w_{x2}\right),\left(t_3,w_{x3}\right),\cdots,\left(t_k,w_{xk}\right),\cdots,\left(t_n,w_{xn}\right)\right) \tag{1}$$

其中，n：维度，特征项的数量；$k\in[1,n]$；

t_k：文档 x 的第 k 个特征项；

w_{xk}：文档 x 的第 k 个特征项的权值；

通过 TF-IDF 算法计算权值，公式如下：

$$w_{xi}=TF\times IDF=tf_{xk}\times idf_k \tag{2}$$

其中，tf_{xk}：词频，指文档 x 中某个特征项t_k出现的次数；

IDF_k：逆文档频率，表示一个特征项在整个文本集合中分布情况，计算方法如下：

$$idf_k = \log\left(\frac{N}{N_k} + 0.01\right) \tag{3}$$

其中，N为文档集合中的总文本数，N_k为文档集合中出现特征项t_k的文档数目。

4.1.3. 文本特征的权值优化

上述 TF-IDF 算法将文档集合作为整体来考虑，不足之处在于不能反映特征项在类间和类内的分布信息对权值计算的影响，故本文对特征项的权值计算方法进行优化，利用 SD-TF-IDF 算法优化权值计算。

特征项的分布信息又称为特征项频率分布的离散度。离散度可分为类间离散度 Di_{ac} (distribution information among classes)和类内离散度 Di_{ic} (distribution information inside a class)两种，分别表示特征项在类间与类内文档间的分布差异。

特征项的类间离散度 Di_{ac}，表述特征项t_k在各类文档中分布的均衡度，公式如下：

$$Di_{ac} = \frac{\sqrt{\frac{1}{m-1}\sum_{i=1}^{m}\left(tf_i(t_k) - \overline{tf(t_k)}\right)^2}}{tf(t_k)} \tag{4}$$

m：文档类别数；

$tf_i(t_k)$：特征项t_k在第 i 类文档中出现的频度；

$\overline{tf(t_k)}$：特征项t_k在各类文档中出现频度的平均值；

$tf(t_k)$：特征项t_k在各类文档中出现的总频度。

特征项的类内离散度 Di_{ic}，表述特征项t_k在某类文档中分布的均衡度，公式如下：

$$Di_{ic} = \frac{\sqrt{\frac{1}{n-1}\sum_{j=1}^{n}\left(tf'_j(t_k) - \overline{tf'(t_k)}\right)^2}}{tf'(t_k)} \tag{5}$$

n：第 i 类中文档总数；

$tf'_j(t_k)$：特征项t_k在第 i 类第 j 篇文档中的出现频度；

$\overline{tf'(t_k)}$：特征项t_k在第 i 类各篇文档中出现频度的平均值；

$tf'(t_k)$：特征项t_k在第 i 类各篇文档中出现的总频度。

在数学中非线性函数 $f(x) = x/(1+x)$ 和 Sigmiod 函数 $f(x) = 1/\left(1+e^{-x}\right)$ 具有良好的收敛性和稳定性。但相对于非线性函数而言，Sigmiod 函数的收敛性和稳定性更好。因此，本文结合上述类内、类间离散度公式，将传统的 TF-IDF 公式进行 Sigmiod 函数运算，优化后的算法公式为 SD-TF-IDF：

$$w_{xi} = \frac{Di_{ac} \times (1 - Di_{ic})}{1 + e^{-(tf_{xk} \times idf_k)}} \tag{6}$$

当特征项分布相对均匀时，其权值较大；反之，其权值较小。

4.2. 文本相似度计算

文本间的相似度是通过使用空间向量余弦来计算的，两个文档的相似度是通过两个向量夹角的余弦值来对比，余弦相似度比距离度量更加强调两个向量方向上的差异[11]。余弦相似度的计算公式如：

$$\text{similarity}\left(\vec{x},\vec{y}\right)=\cos\theta=\frac{\vec{x}*\vec{y}}{\|x\|*\|y\|} \tag{7}$$

文档 x 与文档 y 的空间向量分别为：$\vec{x}$ 与 $\vec{y}$。

因余弦相似度是从方向区分差异，缺少绝对值的敏感度，因此通过调整余弦相似度在所有维度数值上都减去一个均值来修正余弦值无法衡量维数差异的问题[12]。调整余弦相似度的公式如：

$$\text{similarity}'\left(\vec{x},\vec{y}\right)=\cos\theta=\frac{\sum_{i=1}^{n}\left(w_{xk}-\overline{w_x}\right)\left(w_{yk}-\overline{w_y}\right)}{\sqrt{\sum_{i=1}^{n}\left(w_{xk}-\overline{w_x}\right)^2}\sqrt{\sum_{i=1}^{n}\left(w_{yk}-\overline{w_y}\right)^2}} \tag{8}$$

n：维度；

w_{xk}：文档 x 的第 k 个特征项的权值；

$\overline{w_x}$：文档 x 所有特征项的平均数；

w_{yk}：文档 y 的第 k 个特征项的权值；

$\overline{w_y}$：文档 y 所有特征项的平均数。

4.3. 文本聚类步骤

在文档集合中任意选取一篇没有经过处理的文档 p，然后对文档 p 进行标示已经处理，文档处理流程见图 1 所示。具体步骤如下：

1) 对文档 p 进行特征提取，并在提取的过程中对特征进行去噪处理，得到特征集合。

2) 使用 SD-TF-IDF 算法对文档集合中的所有文档进行特征集合的权值优化，得到一个 n 维特征向量。

3) 利用文本相似度算法计算文档 p 与文档集中其他的文档的相似度，假如和文档 p 相似度大于或等于设定阈值，就将存放在文档 p 的领域中，等着特征向量所有的数据全部处理结束。

4) 根据文档 p 领域中的文档数量，与 Minpts 值进行比较，假如大于 Minpts 设定值，则 p 为核心对象，继续往下执行，否则就需返回上一层，从文档集合中再抽取一篇未被处理过的文档，重新对新抽取文档进行处理。

5) 判断文档 p 是否属于某个簇，假如文档 p 没有属于那个簇，再建一个簇，并将文档放入这个簇中。

6) 从文档 p 领域中抽取一篇新的未处理多的文档 r，并将标记文档 r 已被处理，对文档 r 得到领域的方式与文档 p 处理方式相同，判断 r 领域中文档的数量，是否大于 Minpts 设定值，大于就将文档 r 领域中的文档存放在文档 p 的簇中，一直将 p 领域的文档都处理完为止。

7) 文档集中的所有文档都已经处理完，算法结束。

5. DBSCAN 优化算法在实验文本大数据分析中的应用

5.1. 实验

高校计算机实验在其组织、运行、实施过程中会产生丰富的数据。其中，文本数据处理具有一定难度。本文基于 Hadoop 和 Spark 混合框架设计并实现了一个面向高校计算机实验教学的大数据管理平台，用于采集、处理、分析和挖掘实验大数据。按照图 1 所示流程，首先，随机抽取 464 篇文档，包括面向对象程序设计文档相关文档 44 篇，数据结构与算法相关文档 74 篇，数据库原理及应用相关文档 76 篇，大数据技术相关文档 270 篇；然后，分别对每一个类别的文档进行预处理后，利用本文所提 DBSCAN 优化算法进行文本聚类，一共进行了 8 次聚类，结果见表 1 所示。

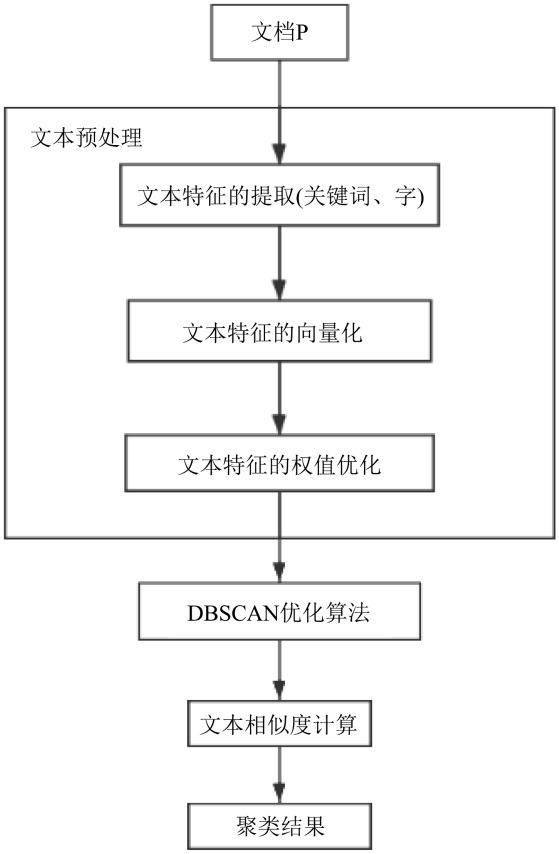

Figure 1. Text clustering based on DBSCAN optimization algorithm
图 1. 基于 DBSCAN 优化算法的文本聚类

Table 1. Results of clustering calculation
表 1. 聚类计算结果

文档	文档数量	簇数量	聚类数	聚类正确数	查全率%	查准率
面向对象程序设计文档	44	1	58	36	81.8%	62%
数据结构与算法文档	74	2	96	64	86.4%	66.7%
数据库原理及应用文档	76	2	82	64	84.2%	78%
大数据技术文档	270	5	290	254	94%	87.6%

5.2. 实验结果分析

从表 1 可以看出，文档集中的有些文档没有和其他类的文档聚在一起成为一类文档，而是单独成一些孤立的噪点，并且有些文档通过计算分析后，也没有准确聚到相似的簇中。当然，出现这些问题的因素比较多，例如有些文档中的特征不能很准确地反映出文档的特征，特征词汇对文档间的区分反应也不是很明显。但是，整体上来看，DBSCAN 优化算法的聚类效果比传统算法的聚类效果要好，它可以能够很好将相似的文档聚类，有效地对文档数据进行分类管理。

6. 结束语

本文提出了一种 DBSCAN 优化算法，实现了参数的自适应选择，提高了聚类准确率，且兼顾了算法

的运行效率；同时，设计了一种文本聚类流程。在实验文本大数据的应用中，通过 DBSCAN 优化算法对相似的文档进行聚类分析。实验结果表明，DBSCAN 优化算法能有效地对大部分的文档进行聚类，实现了高校实验文本大数据的智能化管理，借此可以帮助教师完成实验教学文档的整理归类，为调整优化实验课程内容和方法提供可靠依据，有利于提高学生的学习效率，提升老师的教学水平，达到“智慧学”、“智慧教”的目标。

基金项目

教育部科技发展中心高校产学研创新基金——新一代信息技术创新项目(2018A01015)，教育部科技发展中心高校产学研创新基金——新一代信息技术创新项目(2018A02027)，国家自然科学基金项目(61871475，61471133)。

参考文献

[1] Li, Z., Yang, C., Liu, K., *et al*. (2016) Automatic Scaling Hadoop in the Cloud for Efficient Process of Big Geospatial Data. *International Journal of Geo-Information*, **5**, 173. https://doi.org/10.3390/ijgi5100173

[2] 李璐明, 蒋新华, 廖律超. 基于弹性分布数据集的海量空间数据密度聚类[J]. 湖南大学学报(自科版), 2015, 42(8): 116-124.

[3] Ester, M., Kriegel, H.P. and Xu, X. (1996) A Density-Based Algorithm for Discovering Clusters a Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *International Conference on Knowledge Discovery and Data Mining*, Portland, Oregon, 1996, 226-231.

[4] Manyika, J., Chui, M., Brown, B., *et al*. (2011) Big Data: The Next Frontier for Innovation, Competition and Productivity. McKinsey & Company, Taipei, 3-17.

[5] Lohr, S. (2012) The Age of Big Data. *International Journal of Communications, Network and System Sciences*, **16**, 10-15.

[6] 刘远超, 王晓龙, 刘秉权, 等. 信息探索中的聚类分析技术[J]. 电子与信息学报, 2006(4): 29-32.

[7] Khan, M., Jin, Y., Li, M., *et al*. (2016) Hadoop Performance Modeling for Job Estimation and Resource Provisioning. *IEEE Transactions on Parallel & Distributed Systems*, **27**, 441-454. https://doi.org/10.1109/TPDS.2015.2405552

[8] Guo, Y., Rao, J., Cheng, D., *et al*. (2017) iShuffle: Improving Hadoop Performance with Shuffle-on-Write. *IEEE Transactions on Parallel & Distributed Systems*, **28**, 11-20. https://doi.org/10.1109/TPDS.2016.2587645

[9] 许芳芳. 基于 DBSCAN 优化算法的 Web 文本聚类研究[D]: [硕士学位论文]. 上海: 华东师范大学, 2011.

[10] 侯丽利, 董书宝. 基于 NoSQL 数据库的大数据查询技术的研究与应用[J]. 无线互联科技, 2015(1): 147-154.

[11] 傅华忠, 茅剑. 基于 DBSCAN 聚类算法的 Web 文本挖掘[J]. 科技信息, 2007(1): 55-56.

[12] 牛新征, 佘堃. 面向大规模数据的快速并行聚类划分算法研究[J]. 计算机科学, 2012, 39(1): 134-137.

[13] 闫安, 刘琪林. 一种基于参考点的快速密度聚类算法[J]. 微电子学与计算机, 2017, 34(10): 32-35.

[14] 张振亚, 程红梅, 王进, 等. 面向凝聚式层次聚类算法实现的矩阵存储数据结构研究[J]. 计算机科学, 2006, 33(1): 14-17.

[15] 张忠林, 曹志宇, 李元韬. 基于加权欧式距离的 k_means 算法研究[J]. 郑州大学学报(工学版), 2010, 31(1): 89-92.

[16] Hartigan, J.A. (1979) A K-Means Clustering Algorithm. *Applied Statistics*, **28**, 100-108. https://doi.org/10.2307/2346830

[17] 赵慧, 刘希玉, 崔海青. 网格聚类算法[J]. 计算机技术与发展, 2010, 20(9): 83-85.