第 34 卷 第 11 期 计 算 机 工 程 2008 年 6 月
Vol.34 No.11 Computer Engineering June 2008
·博士论文· 文章编号：1000—3428(2008)11—0015—03 文献标识码：A 中图分类号：TP311

基于局部搜索机制的 K-Means 聚类算法

孙越恒，李志圣，何丕廉
(天津大学计算机学院，天津 300072)

摘 要：K-Means 聚类算法的结果质量依赖于初始聚类中心的选择。该文将局部搜索的思想引入 K-Means 算法，提出一种改进的 KMLS 算法。该算法对 K-Means 收敛后的结果使用局部搜索来使其跳出局部极值点，进而再次迭代求优。同时对局部搜索的结果使用 K-Means 算法使其尽快到达一个局部极值点。理论分析证明了算法的可行性和有效性，而在标准文本集上的文本聚类实验表明，相对于传统的 K-Means 算法，该算法改进了聚类结果的质量。

关键词：K-Means 聚类算法；局部搜索机制；KMLS 算法；文本聚类

K-Means Clustering Algorithm Based on Local Search Mechanism

SUN Yue-heng, LI Zhi-sheng, HE Pi-lian
(School of Computer Science and Technology, Tianjin University, Tianjin 300072)

【Abstract】The quality of K-Means clustering algorithm depends on the choice of cluster center. This paper introduces the idea of local search mechanism into K-Means and presents a KMLS algorithm. This algorithm uses the local search mechanism to jump out one local critical point obtained by K-Means, and uses K-Means to quickly find another local critical point. Experiments of text clustering in standard document sets show that this algorithm achieves a better clustering result than the traditional K-Means algorithm does.

【Key words】K-Means clustering algorithm; local search mechanism; KMLS algorithm; text clustering

1 概述

K-Means[1]是一种基于划分的聚类方法，可看作一个组合优化问题，即如何生成使某一目标函数达到最优的划分 P 的问题。虽然K-Means算法以较快的聚类速度、较好的可伸缩性而被广泛采用，但是它的聚类结果依赖于由初始聚类中心出发所遇到的第 1 个局部极值点，因而不同的初始聚类中心对于聚类质量有着较大影响。为此对于K-Means算法的研究主要集中在对初始聚类中心的选择优化上，如文献[2]提出的Buckshot和Fractionation方法，文献[3]提出的RA算法以及文献[4]提出的聚类中心初始化算法。这些算法在一定程度上提高了聚类结果的质量，但是它们仍依赖于K-Means算法的简单启发机制。

本文将局部搜索的思想引入 K-Means 算法，提出了一个基于局部搜索的 KMLS(K-Means based on Local Search)算法。它将局部搜索机制同 K-Means 算法相结合，使用局部搜索的迭代策略来不断改进 K-Means 算法的聚类结果，同时使用 K-Means 来加速局部搜索的收敛。可以证明 KMLS 算法能得到一个比 K-Means 更优的聚类结果。

2 局部搜索机制及其在文本聚类中的应用

2.1 局部搜索机制

局部搜索[5]是解决优化问题的常用技术。它基于贪婪思想，利用邻域函数进行搜索，包括以下步骤：(1)把原问题转化成一个优化问题，即在一个可行域上寻求一个目标函数的最优值。(2)要在可行域中定义邻域结构，即事先指明可行域中每个点的邻域包含哪些点。在定义了邻域关系之后，进行迭代操作，具体可分为 3 步：1)在可行域中选择一个初始点，常用的策略是随机选取；2)确定如何在一点的局部邻域中寻找更优的点，可采用“见好就走”的贪心策略或沿着最大下降方向的梯度策略；3)确定对局部极值点的处理策略。如果一个点的邻域中的所有点都不比这个点更好，那么这个点就是一个局部最优解。

局部搜索虽然简单，但却十分有效，究其原因在于它的重复搜索操作——虽然从某个初始点出发成功的可能性很小，但多次重复之后，却能以很大概率求得问题的解。

2.2 基于局部搜索的文本聚类算法

文本聚类实际上是为了寻找一个使目标函数最优的划分，可以使用局部搜索来进行文本聚类。

为说明方便，作以下定义：

(1) $\boldsymbol{S}=\{\boldsymbol{d}_1,\boldsymbol{d}_2,\cdots,\boldsymbol{d}_N\}$：包含 N 个文本的文本集。

(2) $P=(\boldsymbol{S}_1,\boldsymbol{S}_2,\cdots,\boldsymbol{S}_K)$：文本集 $\boldsymbol{S}$ 的一个 K 划分(即一个聚类结果)，其中，$\boldsymbol{S}=\cup_{i=1,2,\cdots,K}\boldsymbol{S}_i$，$\boldsymbol{S}_i\cap\boldsymbol{S}_j=\varnothing$，当 $i\neq j$ 时。

(3) $n_i=|\boldsymbol{S}_i|$：文本集 $\boldsymbol{S}_i$ 中的文本数，$i=1,2,\cdots,K$。

(4) $\boldsymbol{D}=\sum_{\boldsymbol{d}\in\boldsymbol{S}}\boldsymbol{d}$：文本集 $\boldsymbol{S}$ 中所有文本向量之和。

(5) $\boldsymbol{c}=\boldsymbol{D}/\|\boldsymbol{D}\|$：文本集 $\boldsymbol{S}$ 的中心向量。

2.2.1 问题的定义

文本聚类问题可转化为：在给定文本集 $\boldsymbol{S}$ 的 K 划分集 $\mathbf{P}=\{P=(\boldsymbol{S}_1,\boldsymbol{S}_2,\cdots,\boldsymbol{S}_K)\mid\boldsymbol{S}=\cup_{i=1,\cdots,K}\boldsymbol{S}_i\wedge\boldsymbol{S}_i\cap\boldsymbol{S}_j=\varnothing,i\neq j\}$ 中，搜索使目标函数最优的一个 K 划分。文本聚类中一般采用如下函数作为聚类过程的目标函数：

基金项目：国家自然科学基金资助项目“基于信息几何方法的维数约减和信息抽象模型研究”(60603027)

作者简介：孙越恒(1974－)，男，讲师、博士，主研方向：自然语言处理，网络信息检索与挖掘；李志圣，博士研究生；何丕廉，教授、博士生导师

收稿日期：2007-06-20 **E-mail：**yhs@tju.edu.cn

$$E(P)=\sum_{i=1}^{K}\sum_{\boldsymbol{d}\in\boldsymbol{S}_i}\boldsymbol{d}\cdot\boldsymbol{c}_i \tag{1}$$

2.2.2 邻域的定义

对任意“点” $P=(\boldsymbol{S}_1,\boldsymbol{S}_2,\cdots\boldsymbol{S}_k)\in\mathbf{P}$，任意文本 $\boldsymbol{d}\in\boldsymbol{S}$，$P$ 的一个邻域点可以定义如下：将一个文本从 $\boldsymbol{S}_i$ 移动到 $\boldsymbol{S}_j$ 时得到的新划分 P'，即如果 $\boldsymbol{d}\in\boldsymbol{S}_i$，那么 P 关于文本 $\boldsymbol{d}$ 的邻域为

$$N_d(P)=\{P\}\cup\{P'=(\boldsymbol{S}_1^{'},\boldsymbol{S}_2^{1},\cdots,\boldsymbol{S}_K^{'})\mid \boldsymbol{S}_i^{'}=\boldsymbol{S}_i-\{\boldsymbol{d}\},\ \boldsymbol{S}_j^{'}=\boldsymbol{S}_j\cup\{\boldsymbol{d}\},\boldsymbol{S}_k^{'}=\boldsymbol{S}_k\text{当}k\neq i,j\} \tag{2}$$

2.2.3 搜索策略

本文使用最速下降作为局部搜索策略。它对一个解 P 邻域内的所有解分别做出评估，然后选择那个使目标函数有最大增加的解作为新解。

设 P 的邻域为 $Neighbour(P)$，最速下降搜索策略就是要在 $Neighbour(P)$ 搜索一个满足 $P'=\underset{P,P'\in Neighbor(P)}{\arg\max}\ (E(P')-E(P))$ 的 P'。即对于任意 $P\in Neighbour(P)$ 都存在：

$$E(P')\geq E(P) \tag{3}$$

基于上述定义，在此提出了一种基于局部搜索的文本聚类算法(Local Search Algorithm, LSA)，描述如下：

(1)对一个聚类划分 $P=(\boldsymbol{S}_1,\boldsymbol{S}_2,\cdots,\boldsymbol{S}_K)$；

(2)设置 $max\Delta=0$，$movedDoc=\text{null}$，$target=\text{null}$ (其中，$movedDoc$ 指要移动的文本，$target$ 指要移动到的类)；

对 $\boldsymbol{S}$ 中的每个文本 $\boldsymbol{d}\in\boldsymbol{S}_i$；

对所有的 j，$j=1,2,\cdots,K\wedge j\neq i$，

计算 $\Delta_j E(P)=E(P')-E(P)$；

$b=\max\{\Delta_j E(P)>0\mid j\neq i\}$；

$max\Delta=\max(max\Delta,b)$，$movedDoc=\boldsymbol{d}$，$target=\boldsymbol{S}_j$；

(3)如果 $movedDoc\neq\text{null}$ (已经找到一个最好解)，则将文本 $\boldsymbol{d}$ 从 $\boldsymbol{S}_i$ 移到 $target$，并重新计算 $\boldsymbol{D}_i$ 及 $\boldsymbol{D}_{target}$；

(4)如果得到 $P'=\underset{P,P'\in Neighbor(P)}{\arg\max}\ (E(P')-E(P))$，则退出；否则，返回(2)和(3)。

3 基于局部搜索的 K-Means 算法

LSA 算法可确切地计算出目标函数的变化，但是在每次迭代过程中，目标函数只会增加一个极小的值；而 K-Means 算法则以高效的迭代而著称。本文结合这 2 种算法的优点，提出了一个基于局部搜索的 KMLS 算法。首先，执行一次 K-Means 操作，当 K-Means 收敛后，执行 LSA 算法来跳出 K-Means 所得到的局部极值点，并再次迭代。

KMLS 算法描述如下：

(1)生成一个初始聚类划分 $P=(\boldsymbol{S}_1,\boldsymbol{S}_2,\cdots,\boldsymbol{S}_K)$；

(2)K-Means(P)；

(3)执行 LSA；

(4)如果类 P 没有变化，退出；否则，执行(2)。

该算法结合了 LSA 和 K-Means 的优点。一方面，在每一次局部搜索迭代得到一个解后，K-Means 的迭代优化可以很快得到由这个解出发遇到的第 1 个局部极值点；另一方面，当 K-Means 收敛得到一个局部极值点后，局部搜索的搜索策略又会使 K-Means 从这个局部极值中跳出，以期得到一个更优的解。

4 实验设计与结果分析

4.1 实验设计

本文使用了 3 个标准的文本数据集：Reuters-21578, 20Newsgroups 和 WAP 数据集。在进行聚类实验之前，对所有的数据集都进行了预处理。经过预处理后，各个数据集的统计信息如表 1 所示。

表 1 文本数据集的属性统计表

数据集	类数	文本数	单词数	平均每个文本单词数	平均每个单词的文本频数
Reutes-21578	81	10 377	18 921	44.8	24.6
20Newsgroups	20	18 828	97 820	82.6	15.9
WAP	20	1 560	8 460	138.9	26.6

本文使用熵作为聚类质量的评价指标。设 $C=\{C_1,C_2,\cdots,C_m\}$ 表示聚成类的集合；$K=\{K_1,K_2,\cdots,K_n\}$ 是文本集中真实类的集合，它是由专家手工建立的，作为评价聚类结果的标准答案。C 的熵定义为

$$Entropy(C)=-\sum_{C_j}\frac{n_j}{N}\sum_i p_{ij}\ln(p_{ij}) \tag{4}$$

其中，n_j 是聚成类 C_j 中的文本数；p_{ij} 是类别 C_j 中实际属于真实类 K_i 的概率；N 表示文本集中的文本总数。熵值越小，表示聚类质量越好。

4.2 实验结果与分析

对上述数据集进行预处理后，进行特征选择并生成文本的特征向量，接着以随机方式生成 10 个不同的初始点，然后针对这些初始点分别运行 KMLS 和 K-Means 算法 10 次，并比较这 10 组结果及其平均值。图 1~图 3 分别是在 3 个数据集上运行 2 种算法后的熵值比较。

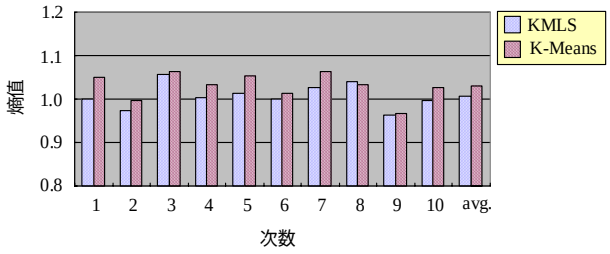

图 1 KMLS 和 K-Means 在 Reuters 上的熵值比较

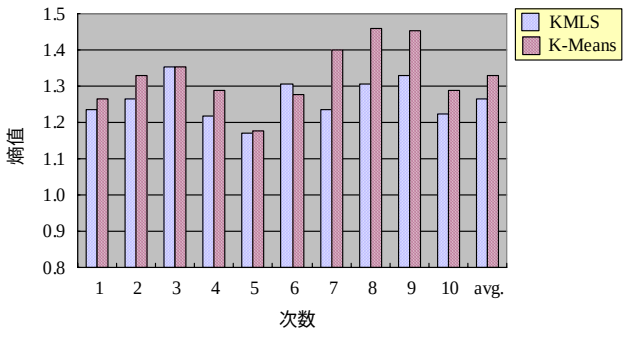

图 2 KMLS 和 K-Means 在 20Newsgroups 上的熵值比较

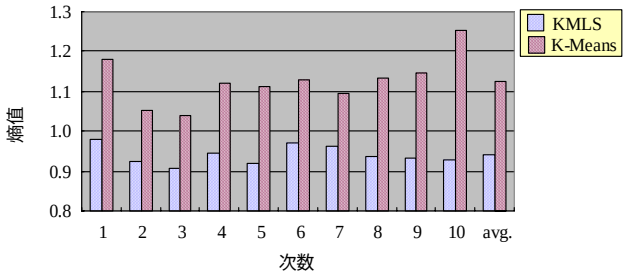

图 3 KMLS 和 K-Means 在 WAP 上的熵值比较

可以看出，KMLS 几乎总是取得比 K-Means 算法更好的结果。注意到在 Reuters 的第 8 次实验和在 20Newsgroups 上的第 6 次实验中，KMLS 所得结果的熵值比 K-Means 要高，为此比较了它们各自的目标函数值，发现 KMLS 所得到结果的目标函数值比 K-Means 要高。之所以会出现这种目标函数值同熵值相矛盾的情况，是因为熵值这个评价标准是以人工分类结果为依据，而向量空间模型在表示文本时，并不能完全地将文本的真实主题表示在余弦相似度的度量之中，因而在某次实验中出现这样的情况是可以接受的。

那么，局部搜索和 K-Means 算法结合的合理性是什么？假定 $P=(\boldsymbol{S}_1,\boldsymbol{S}_2,\cdots,\boldsymbol{S}_K)$，$P'=(\boldsymbol{S}_1',\boldsymbol{S}_2^1,\cdots,\boldsymbol{S}_K')\in Neighbour(P)$，笔者的目标是确定是否可以将一个文本 $\boldsymbol{d}_0\in\boldsymbol{S}_i$ 移动到 $\boldsymbol{S}_j$。

对于 K-Means 算法要检验下面的不等式：

$$\Delta_k=\boldsymbol{d}_0\cdot(\boldsymbol{c}_j-\boldsymbol{c}_i)>0 \tag{5}$$

如果 $\Delta_k>0$，那么 K-Means 把 $\boldsymbol{d}_0$ 从类 $\boldsymbol{S}_i$ 移动到 $\boldsymbol{S}_j$，否则依然把 $\boldsymbol{d}_0$ 留在类 $\boldsymbol{S}_i$ 中。

不同于 K-Means 算法，本文所介绍的 LSA 算法计算：

$$\Delta_{P'}E(P)=E(P')-E(P) \tag{6}$$

对式(6)进一步推导，由 $\boldsymbol{S}_i'=\boldsymbol{S}_i-\{\boldsymbol{d}_0\}$ 和 $\boldsymbol{S}_j'=\boldsymbol{S}_j\cup\{\boldsymbol{d}_0\}$ 可以得到：

$$\begin{aligned}\Delta_{P'}E(P)=&\sum_{i=1}^{K}\sum_{\boldsymbol{d}\in\boldsymbol{S}_i'}\boldsymbol{d}\cdot\boldsymbol{c}_i'-\sum_{i=1}^{K}\sum_{\boldsymbol{d}\in\boldsymbol{S}_i}\boldsymbol{d}\cdot\boldsymbol{c}_i=\\&\sum_{\boldsymbol{d}\in\boldsymbol{S}_i'}\boldsymbol{d}\cdot\boldsymbol{c}_i'-\sum_{d\in\boldsymbol{S}_i}\boldsymbol{d}\cdot\boldsymbol{c}_i)+(\sum_{d\in\boldsymbol{S}_j'}\boldsymbol{d}\cdot\boldsymbol{c}_j'-\sum_{d\in\boldsymbol{S}_j}\boldsymbol{d}\cdot\boldsymbol{c}_j)=\\&\sum_{\boldsymbol{d}\in\boldsymbol{S}_i'}\boldsymbol{d}\cdot\boldsymbol{c}_i'-\sum_{\boldsymbol{d}\in\boldsymbol{S}_i'}\boldsymbol{d}\cdot\boldsymbol{c}_i-\boldsymbol{d}_0\cdot\boldsymbol{c}_i)+(\sum_{d\in\boldsymbol{S}_j'}\boldsymbol{d}\cdot\boldsymbol{c}_j'-\\&\sum_{\boldsymbol{d}\in\boldsymbol{S}_j'}\boldsymbol{d}\cdot\boldsymbol{c}_j+\boldsymbol{d}_0\cdot\boldsymbol{c}_j)=\\&\sum_{\boldsymbol{d}\in\boldsymbol{S}_i'}\boldsymbol{d}\cdot(\boldsymbol{c}_i'-\boldsymbol{c}_i)+\sum_{\boldsymbol{d}\in\boldsymbol{S}_j'}\boldsymbol{d}\cdot(\boldsymbol{c}_j'-\boldsymbol{c}_j)+\boldsymbol{d}_0\cdot(\boldsymbol{c}_j-\boldsymbol{c}_i)\end{aligned}$$

因而：

$$\Delta_{\boldsymbol{P}}E(P)=\sum_{\boldsymbol{d}\in\boldsymbol{S}_i'}\boldsymbol{d}\cdot(\boldsymbol{c}_i'-\boldsymbol{c}_i)+\sum_{\boldsymbol{d}\in\boldsymbol{S}_j'}\boldsymbol{d}\cdot(\boldsymbol{c}_j'-\boldsymbol{c}_j)+\Delta_k \tag{7}$$

由 Cauchy-Schwarz 不等式，有 $\sum_{\boldsymbol{d}\in\boldsymbol{S}_i'}\boldsymbol{d}\cdot\boldsymbol{c}_i'\geq\sum_{\boldsymbol{d}\in\boldsymbol{S}_i'}\boldsymbol{d}\cdot\boldsymbol{c}_i$，因而有：

$$\sum_{\boldsymbol{d}\in\boldsymbol{S}_i'}\boldsymbol{d}\cdot(\boldsymbol{c}_i'-\boldsymbol{c}_i)\geq 0 \tag{8}$$

同样：

$$\sum_{\boldsymbol{d}\in\boldsymbol{S}_j'}\boldsymbol{d}\cdot(\boldsymbol{c}_j'-\boldsymbol{c}_j)\geq 0 \tag{9}$$

由上面 2 个不等式及式(7)可推出：

$$\Delta_{\boldsymbol{P}}E(P)\geq\Delta_k \tag{10}$$

式(8)说明，即使当 $\Delta_k\leq 0$ 时 K-Means 算法不会改变 $\boldsymbol{d}_0$ 在聚类结果内的从属，KMLS 的目标函数 $\Delta_{\boldsymbol{P}}E(P)$ 仍然可能是正的。因此，虽然 $E(P')>E(P)$，K-Means 算法会错过划分 P'。而 2 种算法的结合，则可以避免这种情况的出现，由此改善了聚类结果的质量。

另外，笔者也发现虽然 KMLS 对聚类结果的改善是显著的，但是在大数据集上其改进幅度比在小数据集上要小，如在 Reuters 数据集上，熵值最大的改进是第 1 次运行结果，KMLS 比 K-Means 有 4.46%的改进，而在 WAP 数据集中，最小的改进是第 7 次运行结果，为 11.81%。为进一步验证该结论，手工从 20Newsgroups 数据集中生成了一个每个类包含 50, 100, 200 的小数据集来重复上述实验，结果如图 4 所示。它们的熵值改进分别为：33.93%, 27.12%和 22.88%。

为从理论上说明这个问题，回到式(7)。可以认为，如果把一个文本从一个类移到另一个类时，可以显著地改变这 2 个类的中心概念向量，那么 $\Delta_{\boldsymbol{P}}E(P)-\Delta_k$ 也会有一个较大的改变。当文本集足够大时，由于每个类内文本数较多，因此去除或是加入一个文本对于它的中心概念向量的改变并不显著；然而对于小数据集来说这个改变就非常明显了。

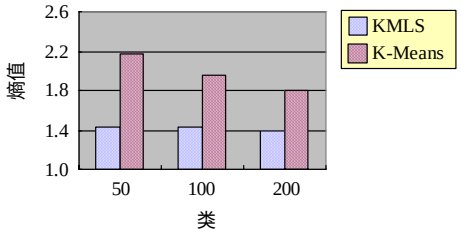

图 4　在 20Newsgroups 小数据集上的熵值比较

与之不同的是，K-Means 算法在面对小数据集时，考虑一个文本 $\boldsymbol{d}\in\boldsymbol{S}_i$，对于任意一个类 $\boldsymbol{S}_j\neq\boldsymbol{S}_i$，由于文本集的文本数很小，而且文本间余弦相似度的均值很小，这就造成了 $\boldsymbol{d}\cdot\boldsymbol{c}_j$ 对于任意的初始类而言都只有很小的数量级。但是对于 $\boldsymbol{d}$ 及它所属的类 $\boldsymbol{S}_i$ 而言，由 $\boldsymbol{d}\cdot\boldsymbol{d}=1$，有

$$\boldsymbol{d}\cdot\boldsymbol{c}_i=\frac{1}{\|\boldsymbol{D}_i\|}+\sum_{x\in\boldsymbol{S}_i-\{\boldsymbol{d}\}}\frac{\boldsymbol{d}\cdot x}{\|\boldsymbol{D}_i\|}$$

上式的第 1 项可以解释为文本 $\boldsymbol{d}$ 对类 $\boldsymbol{S}_i$ 的中心向量的贡献。对于一个小而且高维稀疏的文本数据而言，上面的第一项远大于 $\boldsymbol{d}\cdot\boldsymbol{c}_j$，因而，K-Means 几乎不会在初始类之间移动文本。

5　结束语

本文分析了 K-Means 算法在处理文本数据上的不足，将局部搜索的思想引入文本聚类，提出了一种新的 KMLS 文本聚类算法。

实验结果表明，相对于较大的数据集，KMLS 算法可以更显著地改进小数据集的聚类质量。笔者认为，除了文本表示模型的问题外，另一个原因可能是定义的领域对于目标函数的扰动并不敏感。如何使该算法更好地应用于真实情况下的大规模数据集，有待于进一步研究。

参考文献

[1] Garcia-Escudero L A, Gordaliza A. Robustness Properties of K-Means and Trimmed K-Means[J]. Journal of the American Statistical Association, 1999, 94(8): 956-969.

[2] Cutting D, Karger D, Pedersen J, et al. Scatter-gather: A Cluster-Based Approach to Browsing Large Document Collections[C]//Proc. of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Copenhagen, Denmark: [s. n.], 1992: 318-329.

[3] Larsen B, Aone C. Fast and Effective Text Mining Using Linear-time Document Clustering[C]//Proc. of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Diego, CA, USA: [s. n.], 1999: 16-22.

[4] Khan S S, Ahmad A. Cluster Center Initialization Algorithm for K-Means Clustering[J]. Pattern Recognition Letters, 2004, 25(12): 1293-1302.

[5] Gill P E, Murray W, Wright M H. Practical Optimization[M]. [S. l.]: Academic Press, 1997.