第44卷第10期 中国科学技术大学学报 Vol. 44, No. 10
2014年10月 JOURNAL OF UNIVERSITY OF SCIENCE AND TECHNOLOGY OF CHINA Oct. 2014

文章编号:0253-2778(2014)10-0844-09

视觉特征空间中大规模聚类问题的一种鲁棒近似算法

李大瑞[1],杨林军[2],华先胜[3],张宏江[4]

(1. 中国科学技术大学教育部-微软联合实验室,安徽合肥 230027;2. 微软必应,美国贝尔维尤市 98004;
3. 微软研究院,美国雷德蒙市 98052;4. 金山软件,北京 100085)

摘要:视觉特征空间中的大规模聚类问题是图像识别和检索中亟待解决的问题.当前最好的算法是近似 k-means 算法,它是 Lloyd 算法的近似算法,只能依靠采用高准确率的近似搜索近似地保证聚类结果的性能.为此针对近似 k-means 算法提出改进的基本不增加时间、空间代价新算法,具有更好的算法收敛性和聚类性能.该算法利用了迭代求解过程中更多的信息,更有效地更新子类划分,使得聚类损失单调不增并且快速减小.理论证明,采用任意准确率的近似搜索,该算法都可以在有限轮迭代后收敛到 Lloyd 算法的收敛解.实验结果表明,分别采用最优参数产生同等性能结果时,所提出的算法比近似 k-means 算法快 10 倍.此外,通过比较全局特征聚类实验中的子类的图像,也直观地验证了其聚类效果.

关键词:大规模聚类;k-means;近似算法

中图分类号:TN911.73/TP391.41 **文献标识码**:A doi:10.3969/j.issn.0253-2778.2014.10.008

引用格式:Li Darui, Yang Linjun, Hua Xiansheng, et al. A robust approximate algorithm for large-scale clustering of visual features[J]. Journal of University of Science and Technology of China, 2014, 44(10):844-852.

李大瑞,杨林军,华先胜,等. 视觉特征空间中大规模聚类问题的一种鲁棒近似算法[J]. 中国科学技术大学学报,2014,44(10):844-852.

A robust approximate algorithm for large-scale clustering of visual features

LI Darui[1], YANG Linjun[2], HUA Xiansheng[3], ZHANG Hongjiang[4]

(1. *MOE-Microsoft Key Lab, USTC, Hefei* 230027, *China*; 2. *Microsoft Bing, Bellevue* 98004, USA;
3. *Microsoft Research, Redmond* 98052, USA; 4. *Kingsoft, Beijing* 100085, *China*)

Abstract: The large-scale clustering problem of visual features is crucial for image recognition and retrieval. The state-of-the-art algorithm, the approximate k-means, approximately guarantees the clustering performance by applying the high-precision approximate search. An improved algorithm was proposed, which requires no extra memory cost and nearly no extra time consumption. The robust approximate algorithm can better guarantee its convergence and clustering performance by utilizing more information in the iteration to update the partition, so that clustering loss is non-increasing and reduced rapidly. Theoretical proofs guarantee that the algorithm converges to the converged solution of the Lloyd algorithm, regard less of the precision values of approximate search. The experiment results show that the

收稿日期:2013-09-13;**修回日期**:2014-05-16

作者简介:李大瑞(通讯作者),男,1986 年生,博士生. 研究方向:多媒体信息处理. E-mail: lidarui@mail.ustc.edu.cn

algorithm has about 10 times the speed of the approximate k-means algorithm. Besides, the clustering performance is also directly verified by comparing the images in the clustering results of global features.
Key words: large-scale clustering; k-means; approximate algorithm

0　引言

聚类是图像识别、分类和检索领域中一项广泛应用的技术. 近年来,随着图像识别和检索任务越来越复杂、处理的图像数目越来越多,聚类成为一个亟待解决的问题. 常见的聚类问题有两个:一是产生图像表示时聚类生成视觉词表视觉词表,大规模图像检索系统需要大规模词表[1-2];二是物体发现中聚类图像得到物体子类,大规模实际图像中存在大量物体[3-5]. 不论采用子类原型(prototype)还是采用子类集合作为子类的表示,上述聚类需求都要求聚类算法能够使用的海量样本快速产生大量的子类,本文称之为大规模聚类问题.

视觉特征空间中的大规模聚类问题面临的困难,除了海量样本、大量子类外,还有高维特征. 样本数目至少上千万量级[1-2, 6];聚类数目通常上百万量级[1-2];视觉特征空间维度通常很高,通常接近一百维,最高可以达到几百维,如 128 维 SIFT[1-2],364 维的 GIST[6],使用降维算法后特征维度仍然很高.

上述三个方面的困难对聚类算法提出了巨大的挑战. 传统的聚类算法,即使简单的 Lloyd 算法,它一轮迭代的时间也远远超出实际的承受范围.

当前可行的算法是层次 k-means 算法[1]和近似 k-means 算法[2]. 前者将大规模聚类问题分解成很多嵌套的小规模聚类问题,再分别使用 Lloyd 算法聚类;后者采用近似搜索取代 Lloyd 算法的精确搜索. 这两种算法都极大降低了算法复杂度,但层次 k-means 算法会造成较大的性能损失,而近似 k-means 算法可以通过提高近似搜索的准确率使性能接近 Lloyd 算法.

当前最好的算法是近似 k-means 算法,但它缺乏对收敛性及解的性能的理论保证,只有采用较高准确率时才能保证算法得到较好的解,这同时也限制了算法的执行速度.

本文研究如何减少 k-means 的近似算法与 Lloyd 算法之间的性能差距,减少它对高准确率的近似搜索的依赖. 基本思想是,一方面令聚类损失单调不增,保证算法的收敛性,另一方面使算法在迭代过程中可以高效地减少聚类损失,保证算法收敛速度和聚类性能. 具体的改进思路是,利用近似 k-means 算法迭代过程其他信息来减少近似搜索带来的误差.

利用子类划分在更新前后的相关性和迭代过程中近似搜索的互补性,本文提出了鲁棒近似 k-means 算法(RAKM). 理论分析表明,RAKM 算法会在有限轮收敛到 Lloyd 算法的收敛解. 通过实验,我们验证 RAKM 算法的速度约是近似 k-means 算法的 10 倍.

1　问题及已有算法

1.1　k-means 聚类问题和经典算法

k-means 聚类问题是一个经典问题:给定 N 个样本 x_n 及期望的子类数 K,输出 K 个子类原型 z_k 及表示子类划分的 N 个样本子类标签 a_n,使得平均平方误差(mean squared error, MSE)最小. 记实数矩阵 $\mathbf{X}=[x_n]$、$\mathbf{Z}=[z_k]$,整数向量 $\vec{a}=[a_n]$,则最小化的聚类损失为:

$$\mathbf{J}(\vec{a},\mathbf{Z})=\frac{1}{\mathrm{N}}\sum_n \| x_n-z_{a_n} \|^2 \qquad (1)$$

求解 k-means 聚类问题的全局最优解计算复杂度太高,通常使用启发式方法近似求解. 求解标准算法是 Lloyd 算法[7],它交替执行以下步骤,不断优化 $\vec{a}$ 和 $\mathbf{Z}$,直到收敛或满足其他终止条件:

Step 1　固定 $\mathbf{Z}$,计算 $\vec{a}=\min_{\vec{a}}\mathbf{J}(\vec{a},\mathbf{Z})$;

Step 2　固定 $\vec{a}$,计算 $\mathbf{Z}=\min_{\mathbf{Z}}\mathbf{J}(\vec{a},\mathbf{Z})$.

迭代过程中 Lloyd 算法的聚类损失严格单调减,算法在有限轮收敛到"部分最优解",收敛解 $(\vec{a},\mathbf{Z})$ 中的两项互为最优[8]. 部分最优解通常不是全局最优的,并且从 $(\vec{a},\mathbf{Z})$ 所在解空间角度看,它也不是局部最优解,因为在此解空间中没有明确的局部邻域定义.

Lloyd 算法的计算复杂度主要由 Step 1 决定,它的具体求解为 $a_n=\arg\min_k \| x_n-z_k \|^2$,这需要计算很多样本和子类原型之间的距离. 把一次距离运算作为操作,则执行了 I 轮迭代的 Lloyd 算法,计算复杂度是 $O(NIK)$.

1.2　已有的快速算法

在实际面临的聚类问题中,Lloyd 算法的运行时间远远超出实际承受范围. 以文献[2]中 10^6 的视

觉词表为例，即使只采用 10^8 的 SIFT 特征聚类，在现代计算机上单次距离计算的时间约为 10^{-7} s，Lloyd 算法执行一轮迭代也需要2 778 h.

针对上述问题，文献[9]列举了多种加速策略. 其中，降低聚类算法的时间复杂度是研究重点. 其他加速策略，如并行化[10]和下采样[11]，可以和它一起使用，共同减少算法运行时间. 当前已有的快速算法是层次 k-means 算法和近似 k-means 算法.

层次 k-means 算法（hierarchical k-means，HKM）是一种分层聚类算法. 它采用 Lloyd 算法将输入样本划分成子集，然后利用相同的策略不断细分，得到一个树状结构，将叶子节点中的样本集合作为聚类的子类. 假设树状结构的分支数为 M，共 L 层，则聚类的子类数目 $K=M^L$. 消去 L 并忽略 M 对应的系数，则分支节点迭代轮数为 I 的层次 k-means 算法，计算复杂度是 $O(NI\log K)$.

近似 k-means 算法（approximate k-means，AKM），是 Lloyd 算法的近似. 它采用近似搜索取代 Lloyd 算法中的精确搜索. 其中的近似搜索可采用已有的近似最近邻算法. 如 multiple randomized k-d trees[12]和 k-means tree[13]，它们的时间复杂度均为 $O(\log K)$. I 轮迭代的近似 k-means 算法，时间复杂度为 $O(NI\log K)$.

一般来讲，层次 k-means 的速度更快，但其聚类损失与 Lloyd 算法相比显著升高，而近似 k-means 算法采用较高准确率的近似搜索时聚类损失可以接近 Lloyd 算法. 下面通过性能-时间曲线，比较这三种算法. 图 1 采用了相同的样本集（1.68×10^7 的 SIFT 特征）、相同的初始状态，聚类数目都为 10^5；图中只画了 $I=5,10,20,40,50$ 时的结果.

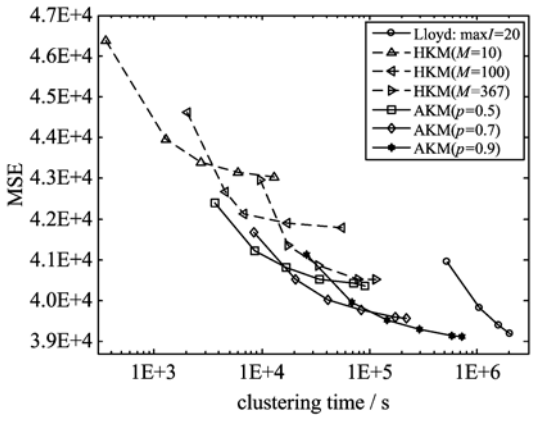

图 1 比较 Lloyd 算法、HKM 算法、AKM 算法

Fig. 1 Compare Lloyd's algorithm, HKM, AKM

实验结果显示，层次 k-means 算法无法取代 Lloyd 算法；而近似 k-means 算法可以取代 Lloyd 算法，采用较高准确率的近似邻搜索，相同性能时它的速度比 Lloyd 算法快很多. 从图 1 可以看出，只有两层（$M=367$）的层次 k-means 算法 HKM，聚类损失与 Lloyd 算法相比也有很大差距，这样的性能差距也无法通过增加迭代轮数减小. 采用较高准确率（$p=0.9$）的近似搜索时，近似 k-means 算法 AKM 的聚类损失与 Lloyd 算法越来越接近，增加迭代轮数可以继续减少聚类损失.

2 鲁棒近似 k-means 算法

下面介绍本文提出的鲁棒近似 k-means 算法（robust approximate k-means，RAKM）.

2.1 基本思想

本文研究如何减少已有的快速算法与 Lloyd 算法的性能差距. 因为近似 k-means 算法的性能差距更小，本文研究如何改进它，特别是采用较低准确率的近似搜索时如何减少性能差距.

本文深入分析了近似 k-means 中近似搜索对性能的影响. 图 2 采用与图 1 相同的实验配置，但在迭代轮数相同时比较性能. 采用性能最好的层次 k-means 和 Lloyd 算法作为分析性能的基准，并画出每轮迭代内部各步骤的性能变化.

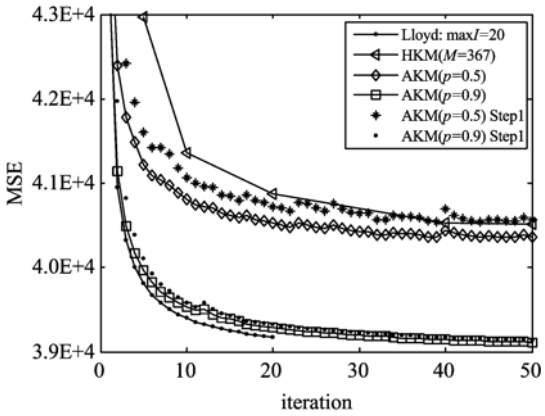

图 2 分析近似搜索的不同准确率对近似 k-means 性能的影响

Fig. 2 Analyze the influences of the approximate searches with different precisions on the performance of AKM

实验结果显示，近似 k-means 算法所采用的近似搜索的准确率会显著影响迭代轮数相同时的算法性能：准确率高才能获得较好性能；准确率低时，迭代不收敛，聚类损失在较高的值附近震荡. 从图 2 可

以看出，近似 k-means 算法 AKM 采用较低准确率（$p=0.5$）的近似搜索时，迭代轮数相同时，性能比最好的层次 k-means 算法 HKM 略好；而准确率较高是（$p=0.9$），性能接近 Lloyd 算法. 近似 k-means 算法经 Step 1 后，聚类损失往往增加；准确率越低，如 $p=0.5$，Step 1 导致的聚类损失增加越显著，这导致聚类损失在一个较大的值附近开始震荡.

上述现象是因为近似 k-means 算法的在 Step 1 中直接使用近似搜索的近似结果更新样本子类标签向量，这样的更新策略引入了近似误差. Step 1 中引入的近似误差导致聚类损失不降反升，因而相邻步骤的聚类损失出现波动，最终，迭代过程中聚类损失在一个较高的值附近震荡，所以减少 Step 1 中引入的聚类误差，是减少近似 k-means 算法与 Lloyd 算法的性能差距的关键.

综上所述，改进的关键在于，对近似 k-means 算法在 Step 1 进行更好的近似划分. 具体的思路包括：①改进的目标. 对迭代过程中聚类损失组成的数列，希望它收敛且收敛到 Lloyd 算法的收敛解的聚类损失. ②改进的方向. 一方面令聚类损失单调不增，另一方面，只要当前的解不是 Lloyd 算法的收敛解，那么迭代过程中它一定会被改进到一个聚类损失更小的解. ③改进的基础. 不同轮迭代的子类划分和子类原型存在着相关性，算法可以利用这些信息帮助生成子类划分.

2.2　改进策略

首先，利用过迭代过程中子类划分的相关性，使得单轮迭代中 Step 1 生成的子类划分不会变差，保证聚类损失单调不增. 其次，利用迭代过程中子类原型的相关性，使得多轮迭代的近似搜索互补，保证算法可以高效地减少聚类损失. 再次，要求相邻轮的近似搜索具有更强的互补性，使得多轮迭代一定能找到所有精确搜索结果，保证收敛到 Lloyd 算法的收敛解.

在 i 轮的 Step 1 中，对 $\mathbf{Z}^{(i-1)}$ 和 $\mathbf{X}$，记精确搜索的结果为 $\vec{e}^{(i)}$，近似搜索的结果为 $\vec{b}^{(i)}$. 记期望产生的子类划分为 $\vec{a}^{(i)}$. 对于近似最近邻算法 Γ，指定准确率 p 和随机种子 r，得近似搜索 $\Gamma(\cdot;\mathbf{Z},p,r)$.

2.2.1　收敛性保证

聚类损失不增，即

$$\mathbf{J}(\vec{a}^{(i)},\mathbf{Z}^{(i-1)})\leqslant\mathbf{J}(\vec{a}^{(i-1)},\mathbf{Z}^{(i-1)}) \tag{2}$$

因为 $\mathbf{Z}$ 是固定的，利用式(1)，令每个样本都满足上述不等关系，得到式(2)的一个充分条件：

条件 1　算法的 Step 1 中，新的子类标签不会比原子类标签和近似搜索的子类标签差，且每个子类标签的改变都会减少聚类损失. 即对 $\mathbf{X}$、$\mathbf{Z}^{(i-1)}$、$\vec{a}^{(i-1)}$、$\vec{b}^{(i)}$，更新的子类划分 $\vec{a}^{(i)}$ 满足 $\forall\, n$，

$$\|x_n-z_{a_n^{(i)}}^{(i-1)}\|^2\leqslant\|x_n-z_{a_n^{(i-1)}}^{(i-1)}\|^2 \tag{3}$$

$$\|x_n-z_{a_n^{(i)}}^{(i-1)}\|^2\leqslant\|x_n-z_{b_n^{(i)}}^{(i-1)}\|^2 \tag{4}$$

当且仅当 $a_n^{(i)}\neq a_n^{(i-1)}$ 时，式(3)取小于.

条件 1 维持一个子类划分 $\vec{a}$，只有减少聚类损失的样本子类标签才能被用来更新它，这样的更新策略使子类划分保持稳定. 聚类损失较大时，已有的子类划分 $\vec{a}$ 往往比 $\vec{e}$ 差很多，$\vec{a}$ 比较容易改进；聚类损失较小时，$\vec{a}$ 中大多数子类的子类标签与 $\vec{e}$ 中的子类标签一致，改进 $\vec{a}$ 困难.

条件 1 使得算法在迭代过程中聚类损失单调不增，从而保证算法收敛.

2.2.2　加速策略

满足条件 1 后，相邻轮的子类划分会越来越相似，那么相应的子类原型矩阵也会很相似，使得相邻轮的近似搜索结果 $\vec{e}$ 有较大的相关性. 这样的情况类似于近似最近邻问题中多次搜索. 在近似最近邻问题中，多次独立搜索可以降低失败概率，如多棵随机化的 k-d 树（multiple randomized k-d trees）[12]. 本文也采用类似的加速策略，得到条件 2.

条件 2　对算法的任意临近的两轮迭代，Step 1 中的近似搜索相互独立，即任意的 $i<j$，精确搜索结果为 $\vec{e}^{(i)}$ 和 $\vec{e}^{(j)}$，近似搜索结果 $\vec{b}^{(i)}$ 和 $\vec{b}^{(j)}$ 满足：

$$P(b_n^{(i)}=e_n^{(i)},b_n^{(j)}=e_n^{(j)})=P(b_n^{(i)}=e_n^{(i)})P(b_n^{(j)}=e_n^{(j)}) \tag{5}$$

满足条件 2 时，在从 i 开始的 m 轮迭代中，对任意样本，未搜到最近邻的概率也随着 m 的增加而显著减小. 若 $i\leqslant j<i+m$，$P(b_n^{(j)}\neq e_n^{(j)})=q$，则 $P(b_n^{(i)}\neq e_n^{(i)},\cdots,b_n^{(i+m-1)}\neq e_n^{(i+m-1)})=q^m$.

同时满足条件 1 和条件 2 时，聚类损失会随着迭代快速减少. 条件 1 使得 $\vec{a}$ 保持稳定，会有很多样本的 e_n 在多轮迭代中保持不变，这时条件 2 可以保证它们找到 e_n 的概率很大，越来越多的 a_n 会改变. 根据条件 1，每个样本标签改变都会导致聚类损失减少，所以条件 1 和条件 2 是相辅相成的，随着迭代增加，条件 2 改进 $\vec{a}$ 的作用越来越大，直到 $\vec{a}$ 在很多轮迭代中都保持不变.

2.2.3　解的部分最优性保证

对于实际的近似最近邻算法，满足条件 1 和条件 2 也不能保证算法一定收敛到 Lloyd 算法的收敛

解. 当前的近似最近邻算法，如 multiple randomized k-d trees 和 k-means tree，都是建立树状结构索引并采用优先队列访问部分子类原型，它们不能保证最近邻一定出现被访问的子集中. 不能保证对任意样本都能找到精确的最近邻，因此近似算法的收敛解未必是 Lloyd 算法的收敛解.

通用算法需要对算法性能的严格理论保证. 为得到一个通用算法，需要多轮迭代中的近似最近邻搜索对任意样本都具有极强的互补性. 下面，我们给出一个确定性的充分条件.

条件 3 对近似最近邻算法 Γ 和任意非零准确率参数 p，存在正整数 m，使得对任意子类原型样本，任意 i 开始的连续 m 个随机种子得出的近似搜索，每个样本的精确子类标签在上述 m 个近似搜索的结果中至少出现一次，即对 $\mathbf{X}$、$\mathbf{Z}$、p、$\vec{e}$，$\vec{b}^{(j)}=\Gamma(\mathbf{X};\mathbf{Z},p,j)$，$i\leqslant j<i+m$ 满足：

$$\vec{e}=\mathrm{comb}(\vec{b}^{(i)},\mathrm{comb}(\vec{b}^{(i+1)},\cdots,\vec{b}^{(i+m-1)}\cdots)) \quad (6)$$

式中，函数 $\vec{b}^{*}=\mathrm{comb}(\vec{b}^{1},\vec{b}^{2})$ 定义为：

$$b_n^{*}=\begin{cases} b_n^2, & \|x_n-z_{b_n^2}\|<\|x_n-z_{b_n^1}\| \\ b_n^1 & \text{otherwise} \end{cases} \quad (7)$$

2.3 具体改进

条件 1 要求，对每个样本，只有减少原子类标签对应的聚类损失的子类标签才能作为新的子类标签，且不能比近似搜索的子类标签差，因此满足条件 1 的最简单的改进如下：

改进 1 Step 1 采用 $\vec{a}^{(i)}=\mathrm{comb}(\vec{a}^{(i-1)},\vec{b}^{(i)})$.

近似最近邻算法通常使用随机化种子产生不同的近似搜索. 近似最近邻算法如何产生独立的近似搜索取决于具体的算法，但它们各自的随机化操作中，通常都会使用伪随机发生器产生的随机数序列构建不同的索引结构，所以随机种子是它们共同的参数. 对伪随机发生器，通常数值接近的随机化种子产生的随机序列会相差很大，这样构建的索引结构，得出的近似搜索也具有较强的不相关性，因此我们把迭代轮数作为随机种子，就可以近似满足条件 2. 条件 2 只是加速策略，不需要严格满足.

改进 2 第 i 轮迭代中，采用 i 作为随机种子产生近似搜索.

条件 3 是对近似最近邻算法一个很强的约束，已有的近似最近邻算法通常并不满足这一约束. 我们给出一个满足条件 3 的改进，它不需要修改近似最近邻算法的实现.

改进 3 对近似最近邻算法 Γ，相应的近似搜索结果为 $\vec{b}=\Gamma(\mathbf{X};\mathbf{Z},p,r)$，令 $\vec{b}^{0}$ 取 $b_n^0=(r \bmod K)+1$，在近似算法中采用 $\mathrm{comb}(\vec{b},\vec{b}^{0})$ 替代 $\vec{b}$.

这三个改进都不需要额外的空间代价；不增加时间复杂度，增加的时间代价比例很小，通常可以忽略. 三个改进除了利用已有的 $\mathbf{Z}$，$\vec{a}$ 两个变量存储迭代的中间状态，不需要额外的空间. 改进 2 不增加时间代价；改进 1 和改进 3 中的 comb 函数使得每个样本增加一次距离计算，在大规模聚类问题中，每个样本的距离计算次数很大，增加的时间代价很小. 以当前最好的近似最近邻算法 k-means tree 为例，近似搜索一个样本，至少要计算它到从根节点到叶子节点的所有的分支中心的距离，距离计算次数一般大于 80.

下面测试近似 k-means 算法 AKM 递增地应用三个改进的效果. 图 3(a)使用图 1 的实验配置，测试改进 1 和改进 2 带来的性能提升；图 3(b)从一个较好的初始状态出发，采用较小的准确率参数 0.3，测试改进 3 增加的距离计算次数对算法时间的影响.

改进后的算法具有更好的收敛速度和聚类性

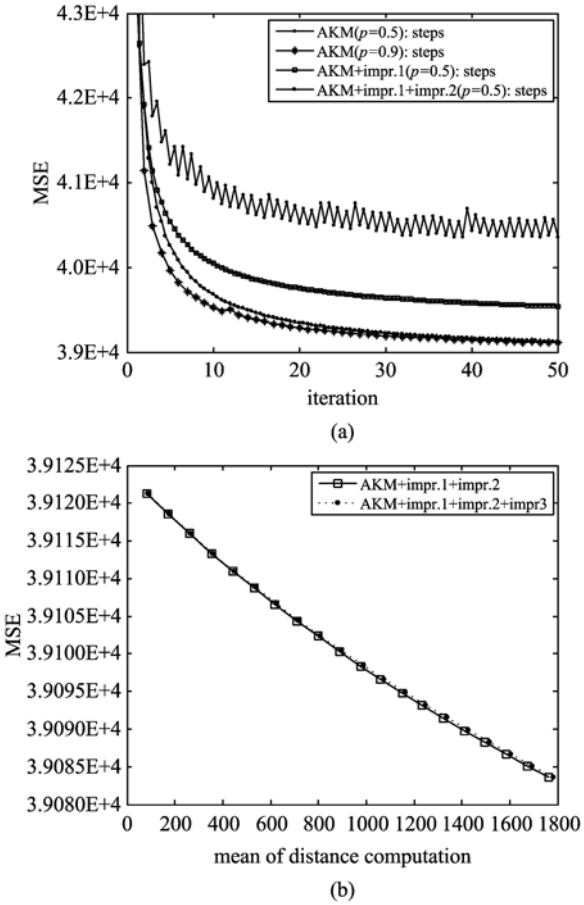

图 3 测试依次增加 3 个改进对近似 k-means 的影响

Fig. 3 Test the influences on AKM by adding the 3 improvements sequentially

能.实验证明,它使得采用较低准确率的算法也能产生高性能的聚类结果,而改进基本不增加时间、空间代价.如图 3(a)所示,准确率为 0.5 的近似 k-means 算法 AKM,使用改进 1 后迭代过程中聚类损失单调不增,迭代轮数相同时的聚类损失大大减小;同时使用改进 2 后,迭代轮数相同时聚类损失非常接近准确率为 0.9 的 AKM.如图 3(b)所示,使用改进 3 前后的时间差别不大.

2.4　RAKM 算法

提出的鲁棒近似 k-means 算法(RAKM)如下:

算法 2.1　RAKM

```
Input: samplesX, cluster numberK, parameterspandI
Output: cluster prototypesZ, partitiona
Initialize Z, a; i=1, r=1
repeat
  Assignment step:
  r=i            /*改进 2*/
  g(·)=Γ(·;Z,p,r)   //Build index using ANN method
  for n = 1 to N do
    b_n=g(x_n)      //Search by approximate search
    b_n=comb(b_n,(rmodK)+1)      /*改进 3*/
    a_n=comb(a_n,b_n)      /*改进 1*/
  end for
  Update step:
  for k=1 to K do
    S_k={x_n | a_n=k}
    z_k=mean(S_k)
  end for
  i=i+1
until i=I OR other termination conditions
return a,Z
```

2.5　初始化和终止条件

对于初始化 RAKM 算法,已有的 k-means 初始化方法面临两方面的难题.①RAKM 算法不仅需要初始化所有子类原型为不同值,还需要给出一个性能较好且与子类原型对应的子类划分.已有初始化方法通常只初始化前者,无法初始化后者.②已有的高性能初始化方法,如 k-means++[14]等,计算复杂度为 O(NK),与 Lloyd 算法的一轮迭代相同,计算时间超出可接受范围.简单的初始化方法,如随机采样方法,性能往往较差.

本文建议使用层次 k-means 算法并采用合适的参数初始化 RAKM.优点包括:①可同时初始化子类原型和子类划分.②它的初始化性能好.③它的时间代价较低.它的时间复杂度代价为 O(NlogK),与近似 k-means 算法的一轮迭代相同,但它的计算时间往往低一个量级,本文对聚类损失没有过高要求.

终止条件也是 RAKM 算法需要注意的问题.若把 RAKM 算法作为通用算法,且收敛是终止条件之一时,判断算法是否收敛不能简单地根据 $\vec{a}$ 或 **Z** 不变.处理高维空间中的大规模聚类问题时,收敛所需的迭代轮数太高,通常把迭代轮数作为终止条件;准确率参数较低时,需要相应地增加迭代轮数.

3　理论分析

3.1　算法收敛性

本文提出的算法在迭代过程中聚类损失单调不增,且在有限轮收敛.

定理 3.1　满足条件 1 的近似 k-means 算法必定在有限轮收敛.

证明　首先,证明收敛性.①算法的聚类损失单调不增.一方面,在第 i 轮迭代的 Step 1,根据式(2)聚类损失单调不增;另一方面,算法的 Step 2 与 Lloyd 算法相同,根据文献[8]有聚类损失单调不增.所以在整个迭代过程中算法的聚类损失单调不增.②算法的聚类损失有界.在有限维欧式空间中,有限个样本,它们的模有界,所以聚类损失有界.综上,有界且单调不增,所以一定收敛.

其次,证明算法在有限轮收敛.①迭代过程中 $(\mathbf{Z},\vec{a})$ 的状态数目有限.由于每轮迭代中 $\vec{a}^{(i)}$ 决定 $\mathbf{Z}^{(i)}$,而 a 是个离散的组合向量,其可行解有限,所以 $(\mathbf{Z},\vec{a})$ 的状态数目有限.②状态转移过程中始终在减小聚类损失,状态转移不可逆.根据条件 1,子类划分发生变化伴随着样本平均误差损失的降低,一定会导致聚类损失减小,所以只要发生状态转移,新状态的聚类损失一定会比旧状态低.③除最终状态外,其他的状态只能持续有限轮,这样的有限个状态的持续轮数存在一个有限的最大值.综合以上三点,算法必定在有限轮收敛到一个不变的状态.

定理 3.1 只能保证算法在有限轮收敛,但它不能保证得到较好的收敛解.

3.2　收敛解的最优性

分析表明,本文提出算法的收敛解是 k-means 聚类问题的部分最优解,与 Lloyd 算法有相同的收敛解.

定理 3.2　采用改进 1~3 的近似 k-means 算法,其收敛解必定是 k-means 聚类问题的部分最优解.

证明　若当前解并非 k-means 聚类问题的部分

最优解，根据条件 3，当前解在 m 轮迭代内一定会被改进，所以算法的收敛解一定是 k-means 算法的部分最优解，它也就是 Lloyd 算法的收敛解.

定理 3.2 显示，本文算法与 Lloyd 算法在性能上没有差距，可以产生 Lloyd 算法的收敛解；只要执行足够多轮，算法一定会获得 Lloyd 算法的收敛解.

4 实验

4.1 实验数据

Oxford Buildings[1] 是一个中等规模的图像检索数据集，它包含5 062张高分辨率(1 024×768)的图像，选择 11 个标志性建筑，每个提供 5 张图像作为测试检索的查询. 所有图像共可得到 1.68×10^7 的 SIFT 特征，特征为 128 维.

实验使用全部的 SIFT 特征聚类，生成不同规模的视觉词表，采用准确率为 0.95 的近似最近邻搜索量化 SIFT 特征得到图像的词袋表示，然后采用向量空间模型得到稀疏的索引结构并进行检索，检索采用 TF-IDF 向量之间的余弦相似度作为排序依据.

Tiny Images[6] 是一个超大规模的场景图像数据集，用于物体发现和图像识别等研究，它包含近 8×10^7 张 32×32 的低分辨率彩色图像. 数据集中的图像是从因特网上收集，同时保存了搜索时的检索关键词等信息. 在这个数据集上，因为只有少量且稀疏的有效标注，所以无法使用纯度或者互信息来评估大规模聚类的性能. 数据集中只检测了 0.03% 的图像和关键词是否相关，相关的也只占 22.9%，远远低于关键词个数.

本文采用 Tiny Images 数据集测试对图像的全局特征聚类以产生大量包含相似图像的子类，并通过观察这些子类图像的视觉外观是否相似来验证大规模图像聚类算法在视觉空间中的有效性.

4.2 算法实现

实验中用到四种聚类算法：Lloyd 算法、层次 k-means 算法、原始的近似 k-means 算法、本文提出的 RAKM 算法.

除了层次 k-means 算法以外的三种聚类算法都使用相同的算法实现框架：先用子类原型建立近似搜索的索引结构，然后再对每个样本搜索其最近子类. 在具体实现中，本文采用文献[13]提供的 FLANN 库① 实现精确和近似搜索. 利用 FLANN 库中的精确线性扫描实现 Lloyd 算法，采用近似最近邻搜索实现近似 k-means 算法与本文提出的 RAKM 算法.

本文实现的层次 k-means 算法是通过修改 FLANN 库中的算法得到. 修改采用贪心策略，在得到指定数目的子类条件下最小化时间代价. 首先，我们取消原有算法中不必要的细分，并修改最下层分指数，使得叶子节点接近指定子类数目；然后，选取样本规模最大的部分叶子节点继续细分，使得叶子节点数目恰好等于指定的子类数目.

4.3 参数配置

近似 k-means 算法和 RAKM 算法需要指定其中的近似最近邻搜索的参数. FLANN 库中存在多种近似最近邻算法，每种算法包含多个参数. 首先，指定近似最近邻搜索的准确度，使用准确度选取时间代价最小的近似最近邻算法和参数. 具体来说，先用 FLANN 库中自动参数配置得出近似最近邻方法的粗略参数，然后再随机采样部分样本进一步估计准确率和事件代价，在满足准确率的前提下选取时间代价最小的方法和参数；其次，对近似最近邻搜索准确率，比较它取不同取值时 RAKM 算法的性能.

实验表明，RAKM 算法的准确率越低越好. 从图 4 可以看出，准确率越低，达到相同聚类损失时的时间代价越小. 这是因为，准确率低时 RAKM 算法可以通过增加迭代轮数的方法达到相同的聚类损失；准确率低的近似最近邻搜索的时间代价大大降低，因而达到相同聚类损失需要的时间代价更小.

本文建议准确率至少为 0.1. 从上述实验结果看，应该用尽可能低的准确率以减少运行时间. 准确

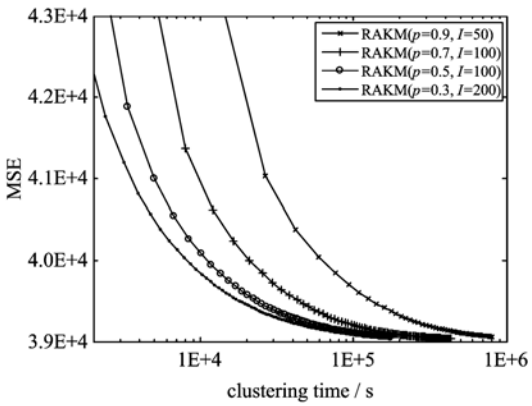

图 4 比较不同准确率时 RAKM 算法时间-损失曲线

Fig. 4 Compare the time-loss curves of RAKM using approximate searches with different precisions

① FLANN: http://www.cs.ubc.ca/~mariusm/index.php/FLANN/.

率太低时，迭代轮数会增加，算法中其他步骤的计算时间会增加，聚类损失减少而速度未必更快. 实验中，通常对 RAKM 算法设置其中的近似最近邻搜索的准确率为 0.5.

本文建议初始化采用层次 k-means 算法，参数使用较小的分支数和迭代轮数. 如果单轮迭代的时间代价较低，可以采用随机采样初始化，通过增加迭代轮数获得较好性能. 当 RAKM 算法单轮迭代的时间代价较大时，采用层次 k-means 算法产生高性能的初始化，可以有效减少迭代轮数，并产生较好的聚类结果. 在保证性能的同时，也要考虑节省计算代价，因为层次 k-means 算法的分支数和节点内的聚类迭代轮数不能太大，所以建议层次 k-means 的参数使用分支数 10、节点内最大迭代轮数 5.

4.4　局部特征聚类实验

本文通过局部特征的聚类实验比较 RAKM 算法和原始的近似 k-means 算法的性能，算法使用各自的最优参数. 实验使用 Oxford Buildings 中全部的局部 SIFT 特征聚类. 原始的近似 k-means 算法准确率参数为 0.9，RAKM 的准确率参数为 0.5.

首先，比较 RAKM 算法和近似 k-means 算法的运行速度. 聚类产生 10^5 规模的视觉词表. 如图 5 所示，在相同聚类损失情况下，近似 k-means 算法相对于 Lloyd 算法的加速比在 10^2 规模，而本文算法的加速比在 10^3 规模；算法速度是近似 k-means 算法的 10 倍以上.

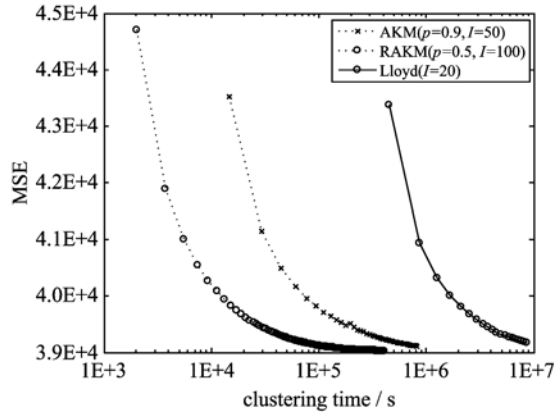

图 5　比较 AKM 和 RAKM 的算法相对于 Lloyd 算法的加速比

Fig. 5　Compare the speedups of AKM and RAKM with respect to Lloyd's Algorithm

其次，比较 RAKM 算法和原始的近似 k-means 算法产生的视觉词表的检索性能和聚类的时间消耗. 聚类数目分别为 10^5 和 10^6. 如表 1 所示，在检索性能近似相等时，RAKM 算法的执行时间约为原始近似 k-means 算法的 1/10.

表 1　比较 AKM 和 RAKM 算法产生近似 mAP 时的时间代价

Tab. 1　Compare the time consumption of AKM and RAKM with similar mAP values

指标	时间(h)		检索性能(mAP)	
词表大小	AKM	RAKM	AKM	RAKM
10^5	200.9	24.2	0.518	0.524
10^6	387.1	30.6	0.608	0.607

上述两个实验表明，RAKM 算法速度约是最好算法(近似 k-means)的 10 倍. 与经典的 Lloyd 算法相比，RAKM 算法的运行时间大约低 3 个数量级.

4.5　全局特征聚类实验

在全局特征聚类实验中测试本文提出的改进和初始化方法所带来的性能提升. 实验从 Tiny Images 数据集下采样 10% 的样本用于聚类，约 7.93×10^6；特征采用 384 维的全局图像特征 GIST.

如图 6 所示，与已有的近似 k-means 算法相比，本文提出的 RAKM 算法在迭代过程可以更有效地减少聚类损失；采用层次 k-means 算法进行初始化时，不仅获得相同的聚类损失迭代轮数更少，而且依照趋势得到聚类结果也好很多. 这样的实验结果表明，在计算复杂度更高的聚类问题中，本文提出的算法和建议采用的初始化方法，不仅可以有效地减少迭代轮数，而且往往会产生更好的聚类结果.

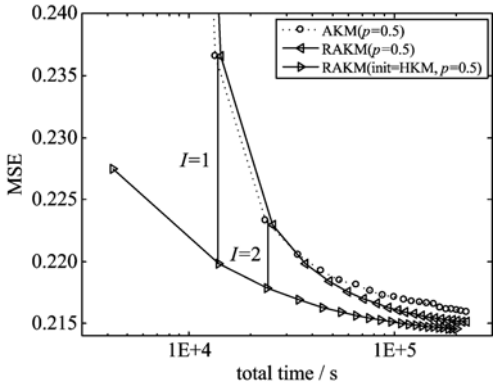

图 6　比较 AKM、RAKM 算法及使用 HKM 初始化的性能

Fig. 6　Compare the performance difference of AKM, RAKM and RAKM initialized by HKM

此外，为了直观显示距离度量和聚类的有效性，随机选取一些子类，给出它们的最近、最远图像.

如图 7 所示，在大多数子类内的图像都有极大的视觉相似性. 子类原型附近图像的相似性更显著，它们

代表子类的共同视觉特征;子类边界附近图像则要杂乱得多,不仅与中心附近图像有较大视觉差异,而且相互之间视觉差异也较大.这样的结果显示了图像视觉外观和 GIST 特征的一致性以及聚类的有效性.

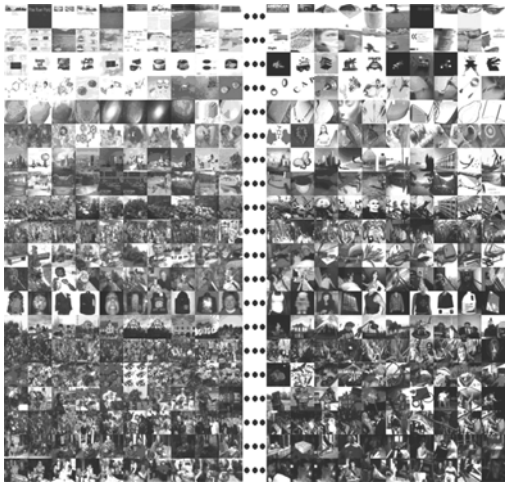

图 7 随机选取的 20 个子类中距离聚类中心最近、最远的 10 张图像

Fig. 7 Show the first and last 10 images close to their clusters prototypes in the 20 random clusters

5 结论

本文针对视觉特征空间中常见的大规模聚类问题,解决如何快速产生高性能聚类结果.本文改进了近似 k-means 算法的子类划分更新,通过使用迭代过程中的更多的信息产生更好的子类划分.本文提出的改进方法,提高了单轮迭代的性能和算法的收敛速度,而且不增加空间代价、几乎不增加时间代价.理论分析表明,提出的 RAKM 算法能在有限轮收敛到 Lloyd 算法的收敛解.实验中,分别选取最好参数,在同等条件下,RAKM 算法速度约是近似 k-means算法的 10 倍,比经典的 Lloyd 算法快 3 个数量级.RAKM 算法聚类局部视觉特征和全局视觉特征都可以快速产生高质量的聚类结果,用在大规模图像检索和物体发现问题中均可获得理想的结果.

参考文献(References)

[1] Nister D, Stewenius H. Scalable recognition with a vocabulary tree[C]// 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. New York, USA: IEEE Press, 2006, 2: 2 161-2 168.

[2] Philbin J, Chum O, Isard M, et al. Object retrieval with large vocabularies and fast spatial matching[C]// IEEE Conference on Computer Vision and Pattern Recognition. Minneapolis, USA: IEEE Press, 2007: 1-8.

[3] Tuytelaars T, Lampert C H, Blaschko M B, et al. Unsupervised object discovery: A comparison[J]. International Journal of Computer Vision, 2010, 88(2): 284-302.

[4] Philbin J, Zisserman A. Object mining using a matching graph on very large image collections[C]// Proceedings of the Inidan Conference on Computer Vision, Graphics & Image Processing. Bhubaneswar, India: IEEE Press, 2008: 738-745.

[5] Li X W, Wu C C, Zach C, et al. Modeling and recognition of landmark image collections using iconic scene graphs[C]// Proceedings of 10th European Conference on Computer Vision. Marseille, France: Springer, 2008: 427-440.

[6] Torralba A, Fergus R, Freeman W T. 80 million tiny images: A large data set for nonparametric object and scene recognition[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2008, 30(11): 1 958-1 970.

[7] Lloyd S. Least squares quantization in PCM[J]. IEEE Transactions on Information Theory, 1982,28(2):129-136.

[8] Selim S Z, Ismail M A. K-means-type algorithms: A generalized convergence theorem and characterization of local optimality[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1984, 6(1): 81-87.

[9] Jain A K. Data clustering: 50 years beyond K-means[J]. Pattern Recognition Letters, 2010,31(8):651-666.

[10] Judd D, McKinley P K, Jain A K. Large-scale parallel data clustering[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1998,20(8): 871-876.

[11] Kollios G, Gunopulos D, Koudas N, et al. Efficient biased sampling for approximate clustering and outlier detection in large data sets[J]. IEEE Transactions on Knowledge and Data Engineering, 2003, 15(5): 1 170-1 187.

[12] Silpa-Anan C, Hartley R. Optimised KD-trees for fast image descriptor matching[C]// IEEE Conference on Computer Vision and Pattern Recognition. Anchorage, USA: IEEE Press, 2008: 1-8.

[13] Muja M, Lowe D G. Fast approximate nearest neighbors with automatic algorithm configuration[C]// Proceedings of the 4th International Conference on Computer Vision Theory and Applications. Lisboa, Portugal: IEEE Press, 2009: 331-340.

[14] Arthur D, Vassilvitskii S. k-means++: The advantages of careful seeding[C]// Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms. Portland, USA: ACM Press, 2007: 1 027-1 035.