DOI: 10.3969/j.issn.1671-0673.2013.03.013

近似最近邻搜索算法——位置敏感哈希

高毫林[1],徐 旭[2],李弼程[1]

(1.信息工程大学,河南 郑州 450001;2.92746 部队,北京 102206)

摘要: 寻找查询点的最近邻是信息处理相关领域的主要任务之一。在数据规模较大时需要采用快速检索算法,常用的快速检索算法主要是基于树的算法,但是当数据点维数较高时,这些算法的效率会变低。位置敏感哈希是当前解决高维搜索的最快的算法,文章对汉明空间、欧式空间下的位置敏感哈希算法的实现方案进行了详细分析,对算法中数据点冲突概率、空间时间消耗、参数调整对算法性能的影响进行了详尽的研究和试验,最后讨论算法的优点和缺点,说明了算法应用于视觉聚类的可能性。

关键词: 近似最近邻搜索; 位置敏感哈希; 精确欧式距离位置敏感哈希; 视觉聚类

中图分类号: TP391.4 **文献标识码:** A **文章编号:** 1671-0673(2013)03-0332-09

Approximate Nearest Neighbor Searching Algorithm—Locality Sensitive Hashing

GAO Hao-lin[1], XU Xu[2], LI Bi-cheng[1]

(1. Information Engineering University, Zhengzhou 450001, China; 2. Unit 92746, Beijing 102206, China)

Abstract: Finding nearest neighbor is a main task in information processing field. The fast searching algorithm is needed in large scale database, and tree-based methods are frequently used for fast retrieval. But when the dimension of data point is high, they will become inefficient. Locality Sensitive Hashing is the fastest method for solving fast high dimension searching currently. This paper explores the implementation of Locality Sensitive Hashing in hamming space and Euclidean space, and studies the data point collision probability, space and time consuming, the effect of parameter tuning through experiments. Finally discussed are the merits and drawbacks of this algorithm and the feasibility of applying LSH in visual clustering.

Key words: approximate nearest neighbor(ANN); locality sensitive Hashing(LSH); exact euclidean locality sensitive Hashing(E^2LSH); visual clustering

0 引言

从海量数据中寻找有用信息是一个很重要的任务。比如需要知道一个图像集上的图像包含了什么内容,如果没有先验信息,很难做到。如果有一个已标注的该图像集的子集,可以通过寻找与已标注图像最相似的图像,并将该标注复制到相似图像来实现。这种方法在现有的大规模图像集上几乎是不可行的,因为不少数据集已经达到了 10 亿数量级。

在这样的数据库上一个基本的计算问题就是最近邻(nearest neighbor,NN)问题:给定一个 n 个对象的

收稿日期: 2013-01-14; **修回日期:** 2013-03-04

基金项目: 国家自科学基金资助项目(60872142)

作者简介: 高毫林(1979-),男,博士生,主要研究方向为图像检索。

集合,在其上创建一个数据结构,对于任意一个查询对象,找出数据集中与它最相似的对象。NN 问题在很多应用领域都有重要的作用。在机器学习中,最近邻规则(nearest neighbor rule) 是一个基本的分类规则。最近邻问题定义如下: 给定一个由 n 个 R^d 空间中的点组成的集合 S,对于任何一个 $q \in R^d$,建立一个数据结构以便快速找到 S 中距 q 最近的点 p。该问题是计算几何的主要问题,由于它的精确解难度大,人们设计了效率更高的近似算法,即近似最近邻算法,又可简化成 R-近邻[1] 。在近似 R-近邻问题中,算法返回与 q 的距离小于 cR 的点(c 大于 1) 。

定义 1　给定一个由度量空间(X,d) 的点组成的数据集和,半径 $R>0$,构建一个数据结构,对于一个查询点 $q \in X$,找到所有满足 $d(v,q) \leqslant cR$ 的点 v。点集 $B(v,r) = \{q \mid d(v,q) \leqslant r\}$,称作以 v 为中心 r 为半径的球 LSH。

大规模高维数据集上近似 R-近邻问题最好的解决方案是位置敏感哈希(locality sensitive Hashing,LSH) ,高维空间的主要特点是,随着维数增加,任意两点之间最大距离与最小距离趋于相等,如图 1 所示。此时,常用的索引方法将难以达到较好的效果。而 LSH 是基于随机映射算法,它将高维向量映射到低维空间,并且以较大的概率使映射前相近的点映射后仍然相近。LSH 虽然采用近似的方法,不保证得出精确的结果,但是它能以较低的代价返回精确的或接近精确的结果。而此前快速索引主要以基于树的方法为主,这些方法在向量维数过高时效率快速下降。以 LSH 为基础的算法有多种[1-6] ,在不同度量空间及不同相似度量条件下有不同的方案。它们在多种场合得到应用,如计算生物学[7] 、物体识别[8] 、音乐识别[9] 、图像检索[10-11] 、音乐检索[12] 、复制检测[13] 、近似重复检测[14] 和名词聚类[15] 等。

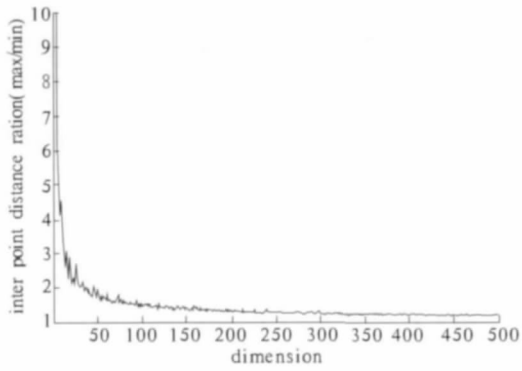

图 1　任意两点之间距离的最大值与最小值之比

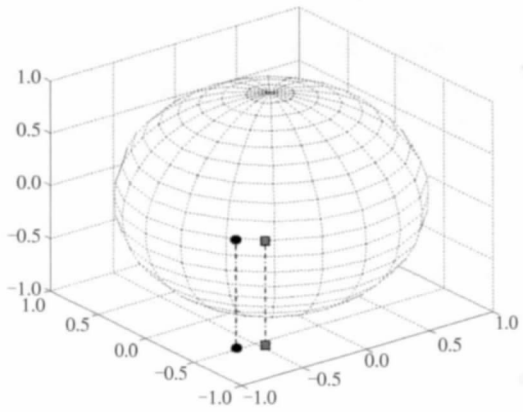

图 2　距离较近的两点映射

1　LSH 基本思想

LSH 的基本思想是: 如果两个点在空间相距很近,那么在进行映射操作后,这两个点仍然相距很近,如图 2 所示。为了对这些点进行映射,先要建立哈希表,可以使一个点的查询在 $O(1)$ 时间内和 $O(N)$ 内存空间上完成查询,N 是数据点的数目。哈希表建立以后,LSH 用一系列哈希函数对数据点进行运算,使那些比较接近的点对于每个哈希函数发生冲突的概率比距离远的点要大,也就是把比较相近的点哈希到同一个桶。这样,通过对查询点进行哈希并获取它所在桶中的标志就可以进一步得到比较近的邻居。对于点集组成的域 S 和距离度量 D,哈希函数 H 将数据点从 S 映射到 U,LSH 的定义如下:

定义 2　一族函数 $H = \{h: S \to U\}$ 被称为(r_1,r_2,p_1,p_2) - 关于度量 D 敏感的,如果对于任何 $q \in S$,

①如果 $v \in B(q,r_1)$,那么 $Pr_H[h(q) = h(v)] \geqslant p_1$,

②如果 $v \notin B(q,r_1)$,那么 $Pr_H[h(q) = h(v)] \leqslant p_2$。

其中,$r_1 < r_2$,$p_1 > p_2$。如果查询点 q 与 v 比较接近,哈希后很可能落入到同一个桶中; 如果 q 与 v 相距较远,落入到同一个桶的可能性较小。因为概率 p_1 与 p_2 的差距可能不够大,需要进行放大。可以将几个哈希函数 $h \in H$ 连接起来。定义一个函数族 $G = \{g: S \to U^k\}$,其中 $g(v) = (h_1(v),\cdots,h_k(v))$,$G$ 中选择独立且分不一致的 L 个函数 $g_1,\cdots,g_L$。在预处理过程中,对于每个 $i = 1,\cdots,L$,算法把每个点 $v \in S$ 存储在桶

$g_i(v)$ 中。因为桶的总数可能很大,算法通过哈希只保留非空的桶。给出查询点 q 后,算法搜索所有的桶 $g_1(q)$,…,$g_L(q)$,并对某个桶中发现的每个点 v,计算 q 到 v 的距离,如果 $\|q-v\| \leqslant cR$(v 是一个 R 近邻),则认为 v 就是算法要得到的点。基本流程如下:

预处理

①选择 L 个函数 $g_j, j=1,\cdots,L, g(j)=(h_{1,j},h_{2,j},\cdots,h_{k,j})$,其中 $h_{1,j},h_{2,j},\cdots,h_{k,j}$ 从 LSH 函数族 H 中随机选取;

②构建 L 个哈希表,对于每个 $j=1,\cdots,L$,第 j 个包含所有使用函数 g_j 哈希的数据集的点。

对点 q 进行查询: 对每个 $j=1,\cdots,L$

①获取第 j 个哈希表中桶 $g_j(q)$ 的点;

②计算每个获取的点与 q 的距离,返回正确的点;

③当返回点数大于 L' 或所有点都经过比较时算法停止。

2 LSH 实现方案

LSH 由文献[1]于 1998 年提出,主要解决汉明空间的高维搜索问题; 2004 年,文献[3]将 p 稳定分布函数引入 LSH,并将 LSH 的使用范围扩展到欧式空间; 2005 年,文献[4]给出了欧式空间 LSH 具体实现方案,并称之为 E^2LSH[4]; 2008 年,文献[5]将 Leech lattice 引入文献[1]的 LSH 方案,将查询时间和内存消耗降到接近于文献[6]中给出的下界; 2010 年,文献[7]对 LSH 相关问题进行了详细描述。

2.1 汉明空间实现方案

文献[1]用 ε-NNS 描述近似近邻搜索问题: 找出查询点 q 的 ε 近似最近邻点(ε-NNS) $p \in P$ 对于 P 中所有点 p' 满足 $d(p,q) \leqslant (1+\varepsilon)\, d(p',q)$。$\varepsilon$-PLEB 定义如下:

定义 3 给定中心位于度量空间(X,d)中的点 $C=\{c_1,\cdots,c_n\}$,半径为 r 的 n 个球,设计一个数据结构对任何查询点 $q \in X$ 返回如下结果:

①如果存在 $c_i \in C$ 和 $q \in B(c_i,r)$,就返回 YES 和满足 $q \in B(c_i',(1+\varepsilon)r)$ 的点 c_i';

②如果对所有 $c_i \in C, q \notin B(c_i,(1+\varepsilon)r)$,那么返回 NO;

③如果对于距 q 最近的点 c_i,有 $r \leqslant d(q,c_i) \leqslant (1+\varepsilon)r$,返回结果不确定。

这样,通过寻找 ε-近似最近邻并比较 r 与它和查询点的距离,就可以用同样的预处理和查询代价借助于一个数据结构将 ε-NNS 问题转化为 ε-PLEB(point location in equal balls)问题。这样的数据结构就是 ring-cover tree,它可以在任意的数据集上找到 ring-sepator 或 cover,从而将数据集 P 分解为小的集合 S_1,…,S_l,对于所有的 i 和某个 $c<1$ 满足 $|S_i|<c|P|$,并且 $\sum_i |S_i| \leqslant b|P|$,$b<1+1/\log^2 n$。这种性质可以很快地将搜索范围从整个数据集 P 缩小到集合 S_i。针对 ε-PLEB 问题,作者提出了 LSH 的概念并用于次线性时间搜索,适用于任何 $p \in$ [1,2] 的范数 l_p,也可用于集合相似度(set resemblance)的衡量,如网页、文档聚类等。

在汉明空间上,用 D 表示两个点之间的汉明距离。对任何 $r, \varepsilon > 0$,LSH 函数 $H=\{h: h(v)=v_i=v|_{\{i\}}\}$ 是$\{r, r(1+\varepsilon), 1-r/d, 1-r(1+\varepsilon)/d\}$ 敏感的,其中 i 从$\{0,\cdots,d-1\}$ 均匀随机抽取,也就是说,h 是沿着坐标 i 的映射,从 H^d 映射到 H。对任何 $\varepsilon>1$,H^d 空间中 ε-PLEB 问题的算法使用 $O(dn+n^{1+1/\varepsilon})$ 的空间和 $O(dn^{1/\varepsilon})$ 的查询时间。这样,函数 $g=(h_1,\cdots,h_k)$ 就等价于 $g(v)=v|_{I_i}$,其中,I_i 是大小为 k 集合,每个元素从$\{0,\cdots,d-1\}$ 随机抽取。如果设置 $r_1=R, r_2=cR$,并且 $p_1=1-R/d, p_2=1-cR/d$,可以得到 $k=\dfrac{\log n}{-\log(1-cR/d)}$,$L=O(n^{1/c})$[3]。

集合 A、B 的相似衡量定义为 $D=\dfrac{|A\cap B|}{|A\cup B|}$。令 S 是 $X=\{1,\cdots,x\}$ 的所有子集,对于 $1>r_1>r_2>0$,$H=\{h: h_\pi(A)=\max_{a\in A}\pi_a, \pi$ 是 X 的排列$\}$。关于 D 的$(r,\varepsilon r)$-PLEB 问题使用 $O(dn+n^{1+\rho})$ 的空间和 $O(n^\rho)$ 的查询代价,$\rho=-\ln r/\ln\varepsilon$。两个字符串 A、B 的 l_1 距离 $D(A,B)=\sum_{i=0}^{m-1}|A[i]-B[i]|$,令 $t=cR, c>1$,定义长

为 M 的模式 P 的 LSH 函数如下:

$$h_u(P)=\left(\left\lfloor\frac{P[0]+d_{u,0}}{t}\right\rfloor,\left\lfloor\frac{P[1]+d_{u,1}}{t}\right\rfloor,\cdots,\left\lfloor\frac{P[M-1]+d_{u,M-1}}{t}\right\rfloor\right),$$

其中,$d_{u,j}$从$\{0,\cdots,t-1\}$随机的均匀抽取,$0<j<M$。

文献 [2] 对查询时间作了一些改进,从 $O(dn^{1/\varepsilon})$ 改进到 $O(dn^{1/(1+\varepsilon)})$,每个桶存储点数从一点扩充到多点,计算位置从主内存扩展到辅助内存。将坐标点从欧式空间转换为汉明空间。假设数据集 P 中任一点 p 的坐标的最大值为 C,那么点 $p=(x_1,\cdots,x_d)$ 可以转换为汉明空间 $H^{d'}$ 中的点 $v(p)=\mathrm{Unary}_C(x_1),\cdots,\mathrm{Unary}_C(x_d)$,其中,$d'=Cd$,$\mathrm{Unary}_C(x)$ 是 x 的二进制表示,比如由 x 个 1 和 $C-x$ 个 0 组成的序列。在进行坐标映射时,首先选取 l 个$\{1,\cdots,d'\}$的子集 $I_1,\cdots,I_l$,然后将向量 p 映射到坐标集 I 上(用 $p|I$ 表示),也就是对于每个 I_j 选取对应位置的坐标值,再将这些值(0、1 比特)连接起来,这就是 LSH 函数的定义。$p|I$ 用 $g_j(p)$ 表示,在预处理阶段,将每个点存储在 $g_j(p)$ 中,作者在存储这些点时,定义每个桶容纳的最大点数为 B,当点数超过 B 时,存储在一个连接在该桶上的新桶。如果哈希表的大小用 M 表示,那么 $M=\alpha\frac{n}{B}$,n 为数据集中的点数,α 为内存使用率。由于哈希桶的个数可能会很多,使用标准哈希进行压缩。假设经过 $p|I$ 映射后的向量为$(v_1,\cdots,v_k)$,哈希函数 $h(v_1,\cdots,v_k)=a_1\cdot v_1+\cdots a_k\cdot v_k \bmod M$,其中 M 是哈希表的大小,$a_1,\cdots,a_k$ 是从 $[0,\cdots,M-1]$ 选取的随机数。

该算法中参数 k 应该使距离查询点较近的点与查询点落入同一个桶的概率最大,而距查询点较远的点与它落入同一个桶的概率最小。当 $k=\log_{1/p_2}(n/B)$ 和 $l=(n/B)^{\rho}$ 时,LSH 可以以至少$\frac{1}{2}-\frac{1}{e}\geqslant 0.132$的概率正确地解决 ε-NNS 问题,$\rho=\frac{\log 1/p_1}{\log 1/p_2}$。该方案虽然可以用于欧式空间,但存在两个问题,①二进制量化的方法引入了误差; ②当坐标值较大时,量化后的字符串会非常长,影响运算效率。

2.2　欧式空间实现方案

前面的 LSH 哈希函数主要针对二进制汉明空间$\{0,1\}^d$中的点。数据存储在磁盘上的时候,它们比基于树的结构在速度上有了很大提高。同时,当需要进行增加和删除操作时,它们也能够扩展到动态数据集上。但有一个很重要的缺陷,即只有输入数据位于汉明空间时,才会达到简单而快速的效果。虽然通过嵌入到汉明空间也能把算法扩展到欧式空间,但这在很大程度上增加了算法的查询时间,也增加了算法的错误率和复杂度。

文献 [3] 提出了针对欧氏空间的快速解决方案,文献 [4] 进一步提出了改进方案 E^2LSH。该算法不需要嵌入就可以直接工作在欧式空间中的点上,它还可以工作在任何 l_p 范数上,$p\in(0,2]$。这被证明当时是唯一的可工作于高维空间中 $p<1$ 情况的 NN 问题解决算法。它继承了 LSH 的两个特点,①很适合于维数很高但稀疏的数据点,尤其是当 d 是向量中非零元素的最大数目时,算法的运行时限会保持不变,该特点是其它空间数据结构所不具有的; ②如果数据满足一定的有界增长特性(bounded growth property),它可以很快找到精确的近邻。对于点 q,$c>1$,令 $N(q,c)$ 代表 q 的 c-近似近邻的数目,如果 $N(q,c)$ 以 c 的函数按次线性(sub-exponentially)增长,并且给定常量因子去近似 q 到它的最近邻居的距离,那么,算法能够以固定概率在时间 $O(d\log n)$ 内找到最近的邻居 v。

该算法使用的 LSH 函数族基于 p 稳定分布函数,p 稳定分布是广义高斯分布,满足广义中心极限定理。稳定分布被定义为归一化独立同分布变量和的极限。p 稳定分布的定义如下:

定义 4　存在 $p\geqslant 0$,对于 n 个实数 $v_1,\cdots,v_n$ 和在 R 上的分布 D 的独立同分布变量 $X_1,\cdots,X_n$,如果 $\sum_i v_iX_i$和$(\sum_i|v_i|^p)^{\frac{1}{p}}X$ 分布相同(其中 X 是分布 D 的随机变量),那么分布 D 被称为 p 稳定分布,记为 D_P。

p 稳定分布的重要性质是稳定性,即具有相同指数的 p 稳定分布的随机变量的线性组合仍然是 p 稳定分布随机变量,在经济时间序列、天体运动、信号处理等领域有着重要的作用。对于 $p\in(0,2]$ 的任何 p 值都存在稳定分布,如柯西分布是 1-稳定的,高斯分布是 2-稳定的。p 稳定分布的密度和分布函数没有封闭形式,但可以通过 $[0,1]$ 上均匀分布的独立变量模拟产生 p 稳定随机变量。

在进行哈希运算时,内积$(a\cdot v)$把每个向量映射到一条实线上。由 p 稳定分布定义可知,两个向量

(v_1, v_2) 投影距离$(a \cdot v_1 - a \cdot v_2)$ 的分布与 $\| v_1 - v_2 \|_p X$ 的分布相同(X 服从 p 稳定分布)。如果能够把实线以合适的长度 w 进行等长分割,并且根据向量被投影到分割后的哪一段为该向量分配一个哈希值,那么这样的哈希函数满足前面位置敏感的描述,如图 3 所示。所以 E^2LSH 的哈希函数为 $h(v) = \left\lfloor \frac{a \cdot v + b}{w} \right\rfloor$,两个向量 v_1, v_2 在上述哈希函数下冲突的概率可以计算。

每个函数 g_i 确定一个哈希表,每个哈希表有多个桶$\{g_i(v) \mid v \in S\}$ 组成。为了便于存储,使用两个哈希函数 $h_1: Z^k \to \{0, \cdots, tablesize - 1\}$ 和 $h_2: Z^k \to \{0, \cdots, C\}$($g_i$ 将数据点从 R^d 映射到 Z^k),h_1 用作通用哈希函数,它决定桶在哈希表中序号,h_2 用来确定桶在链表中的标志。这样可以使用一维数据 h_2 而不是 k 维数据进行查询,将查询代价降到了 $O(1)$。

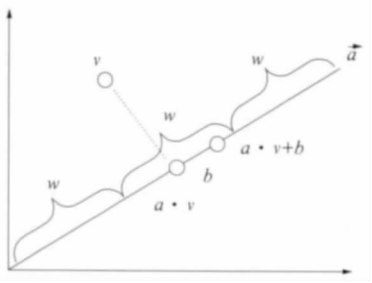

图 3 哈希函数示意图

寻找桶内冲突点可以通过链表解决,当在链表中存储一个桶 $g_i(v) = (x_1, \cdots, x_k)$ 时,不需要存储全部向量$(x_1, \cdots, x_k)$ 作为桶的标志,而只需要存储 $h_2(x_1, \cdots, x_k)$ 的值,不仅可以减少内存的使用量,也便于在表中查找该桶。h_2 范围的选择要足够大,以保证两个不同的桶有不同的标识符。这样桶 $g_i(v) = (x_1, \cdots, x_k)$ 存储的信息就包括 $h_2(x_1, \cdots, x_k)$ 和桶中的点 $g_i^{-1}(x_1, \cdots, x_k) \cap P$。

2.3 算法性能分析

近来 LSH 算法中应用较多的是 E^2LSH,主要对它进行分析。

2.3.1 数据点冲突概率

LSH 的准确度由它发现真实最近邻的概率决定,也就是这两个点冲突的概率,而这个概率又与 p 稳定分布函数有关。设 $f_p(t)$ 代表 p 稳定分布绝对值的概率密度函数,$c = \| v_1 - v_2 \|_p$。两个点冲突的概率 $p(c)$ 的计算见文献[8]的 3.1 节,对于固定的参数 w,$p(c)$ 随 c 单调递减。$p(c)$ 与 w 和 c 的关系如图 4 所示,可见,距离越大的两点映射到同一桶中的概率越小。

值得说明的是,E^2LSH 的随机映射虽然能使距离较近的点哈希后冲突概率较大,但是,即使两个点距离较近,它们冲突的概率并不是足够大的。如文献[8]的 3.2 节所示,E^2LSH 只能保证这个概率大于 1/2。这说明,仅用一个哈希表进行检索是不够的,如果对于准确率有较高的要求,需要增加表的个数。类似的,在进行聚类时,用一个表产生的码本随机性是比较强的,如果直接用于语义检索、近似重复检测和目标识别将很难达到令人满意的效果。这时需要减弱算法的随机性,如从多个表中选取最优者进行融合产生新表、与基于树的方法结合或者与网格的方法结合等。

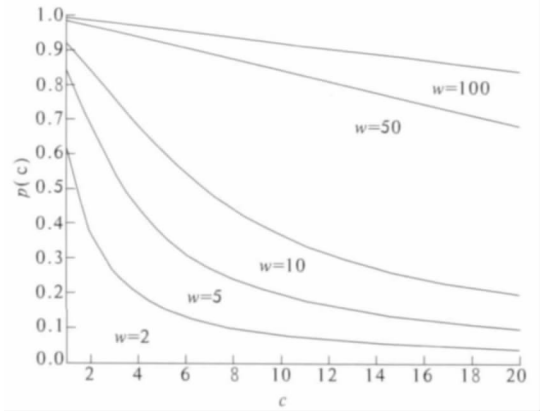

图 4 冲突概率与 w 和 c 的关系

2.3.2 空间时间消耗

查询时间主要包括计算哈希映射的时间 T_g 和寻找冲突点的时间 T_c,$T_g = O(dkL)$,d 是原始数据点的维数,$T_c = O(nLN_c)$,n 表示每个桶中的平均布点数,$N_c = \sum_{p \in P} p^k(\| q - p \|)$ 表示单次映射冲突点数的期望值。可见,T_g 随着 k 的增加而增加,T_c 随着 k 的增加而减少。

ρ 对空间和时间消耗的大小起着重要的作用,它随 w 和 c 的变化如文献[8]的 5.1 节所示。当 c 增大时,ρ 的值不断减小。而 w 只在接近于 1 的较小区间内对 ρ 有明显的影响,这个区间与数据集中点的距离有关。

2.3.3 参数调整对算法影响

在 E^2LSH 算法应用中,当 w 指定时,对算法性能影响较大的参数就是 k 和 L。冲突概率都随 L 的增大而增大,随 k 的增大而减小。增加 w 虽然会增加落入每个桶中的点数,从而增加冲突概率。但是,为了得到最近邻结果需要搜索所有与查询点落入同一个桶中的点,这样就会增加查询时间。详细说明见文献

[18] 的 3.2 节。这样,可以通过增加 L 来提高发现真实近邻的概率,但运算量也会显著增加。文献 [9] 在 cover-song 试验中使用的 k 值为 7 ~ 14,L 大于 150[9]。

3 关于 LSH 算法的讨论

3.1 国内研究现状

国内对 LSH 的相关研究并不多: 文献 [19] 于 2002 年最早介绍 LSH,并将其应用于汉明空间图像检索; 文献 [20] 于 2006 年针对不同数据集的统计特征选取适当的散列函数,实现汉明空间 LSH 方案索引参数的自动调整; 文献 [21] 将汉明空间 LSH 用于中文文本快速检索; 文献 [22] 将欧式空间 LSH 方案用于视频帧的匹配,实现基于内容相似的重复视频片断检测; 文献 [23] 借用汉明空间 LSH 的框架,利用随机子向量设计不同的哈希函数,实现对图像高维特征的匹配; 文献 [24] 于 2011 年将 KD-tree 和汉明空间 LSH 结合进行索引,实现在多核集群架构上并行索引。然而,上述研究还不够深入,而且多是对汉明空间实现方案的应用。

3.2 算法的主要优点及局限

LSH 算法的主要优点是用于检索速度很快,相比于 KD-tree 等算法可以有几十倍的提高; 适合于动态数据集增量索引[25],索引结构更新的计算代价小; 用于聚类时不需要对全体数据集重新聚类生成新码本。主要局限是耗费内存空间较大: 近邻点可能分布在多个桶,如果要达到较好的性能,可以建立多个哈希表,但这种做法内存开销很大。解决方法主要有文献 [26] 提出的 Multi-probe LSH 从一个表中多个桶查找近似点,可以减少约 90% 的空间消耗; 另外,算法性能对参数非常敏感,这些参数必须在使用前加以确定; 文献 [27] 提出的 LSH Forest 减少了需要确定的参数的个数,部分解决了这个问题; 文献 [28] 设计了自适应的 LSH 搜索算法模型,动态地为每次查询确定参数。LSH 的改进还包括文献 [29] 中提出了基于 LSH 的适用于任意核函数的快速搜索技术,增加了在大规模图像库上进行搜索的可能性; 文献 [30] 中将 LSH 引入度量学习,为训练样本学习马氏距离的参数矩阵。

3.3 在视觉聚类中的应用

视觉词袋法(bag of visual words,BOVW) 在场景分类[31-32]、视频搜索[33-34]、物体检测与识别[35-36]等方面的应用逐渐增多。它借用了与文本领域的词袋法(bag of words, BOW)。主要流程如下: 先提取训练图像局部特征,然后采用聚类算法如 K-Means、H-KMeans、A-KMeans 等对特征进行聚类,每个聚类中心看作一个视觉单词(visual word),聚类结果构成视觉词表(visual vocabulary)或视觉码本(visual codebook); 然后将查询图像或待检测、识别图像的各个特征映射到最近的视觉单词,得出一个视觉单词频次直方图,该直方图就代表这幅图像; 最后利用图像词频直方图结合空间金字塔核匹配等方法进行搜索,或者利用图分割、训练分类器等方法进行检测、识别、分类等。

视觉码本是视觉词袋法的关键,它对应于对局部特征聚类的结果,其中的视觉单词表示该类内关键点共有的局部模式,类的数目就是视觉码本的大小。码本的大小与视觉词袋法的性能直接相关。小码本由于分类的数目少,可能使不相似的两个关键点分配到同一个类,从而使码本的区分力(discriminative power)降低。而大码本缺乏归纳性,对噪声容忍能力差,还会增加计算开销。产生视觉码本的聚类就是视觉聚类。常用的视觉聚类方法之一是 K-Means,但 K-Means 存在一些缺点。主要包括: ①聚类中心分布不均导致视觉单词的同义性; ②偏远点使聚类中心漂移产生视觉单词多义性; ③计算效率随数据点增大迅速降低; ④不支持动态扩展,当数据集规模扩大时需要重新聚类。因此,可以考虑用 LSH 聚类,它通过映射将相似数据点分配到同一个桶,将这样的桶作为一个类也可以达到对数据聚类的效果。图 5 给出了 K-Means 与 LSH 对同一组数据进行聚类的结果。可见,LSH 聚类在数据点稠密区得出的类数目少,归纳性强于 K-Means; 同时,在数据点稀疏区得出的类数目多,区分性强于 K-Means。这与一些文献 BOVW 加权方案中重视信息量少的点,给这些点分配较大的权值的思想类似。

虽然有文章指出内存消耗大是 LSH 最大局限[11,37-38],但通过将哈希运算的各个桶分配结果存为独立的外存文件,检索时先计算查询点所对应的桶标志,就可以把搜索范围缩小到相关的桶文件,大大降低内存消耗。这一点已经实验中得到了证实。同时,LSH 提出者曾经指出可以将 LSH 用于快速聚类[1],尽管

他并没有用实践证明,但在文本聚类中已有应用[15,39]。虽然在视觉聚类中由于视觉单词具有不确定性,这使得它比文本聚类要复杂,但也有相似之处。因此,将 LSH 用于视觉聚类具有可行性。文献[40]已有这方面的尝试,文献[41]也用 LSH 实现局部特征到视觉单词的映射。由于随机性的存在,将 LSH 用于视觉聚类要降低它产生的码本的随机性。文献[40]指出可以使用 ERC-Forest[42]降低表哈希结果的随机性,但并没有详细描述。另外,强调 LSH 内存消耗大的文献指出的问题是 LSH 在检索中的应用,而不是在聚类中的应用。

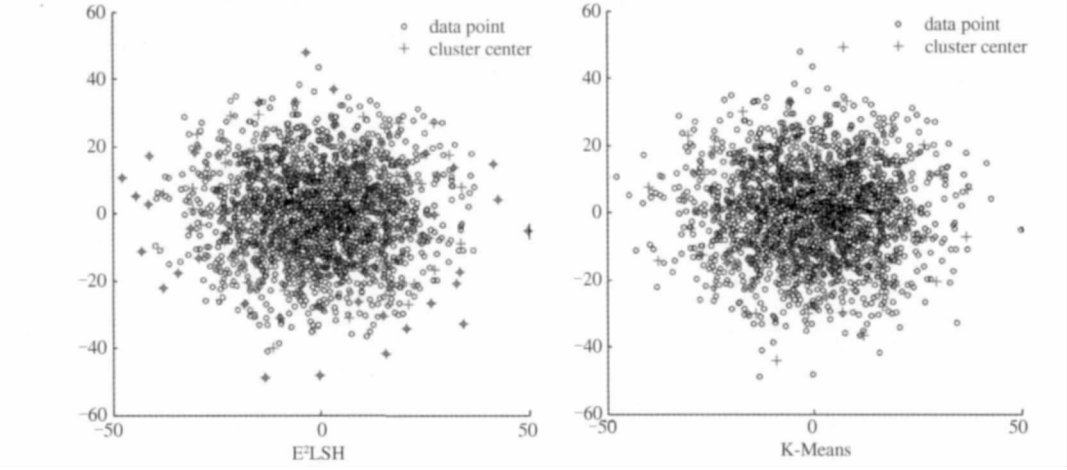

图 5 K-Means 与 E^2LSH 对同一组数据聚类结果

应该说明的是,将 LSH 用于视觉聚类需要克服的主要问题是它的随机性。用单个哈希表的随机性较强,难以达到令人满意的效果,而使用多个哈希表聚类又存在如何对聚类结果进行融合的问题。该问题的解决需要增加计算代价,但只要方法合适,就可以达到效率和聚类质量的折衷,而且聚类可以离线进行,对速度要求不高。另外,LSH 方法提供的是一个框架式方法,它的哈希函数可以根据需要确定,如用距离、格形编码等。

4 结束语

哈希运算在信息检索等领域有着广泛的应用,位置敏感哈希是近几年应用最多的基于随机映射的哈希算法,尤其是在用高维向量表征的大规模数据库上取得了比较好的效果。因为在维数较高时常用的基于树的算法查询时间会随着数据规模的增大而迅速增大,而 LSH 算法则会在保证一定的检索精度时使检索时间大大减少。在计算机视觉领域,LSH 已应用于用全局特征和关键点特征表示的图像检索,并且取得了较好的效果。除了用于检索以外,正如作者指出的,位置敏感哈希还可用于快速聚类,这在文本处理中已有应用。而在视觉词典法中,如果对它进行改进并应用于视觉聚类,也存在一定的可能性,因为它能克服当前视觉聚类主要使用的 K-Means 算法存在的缺点。但同时由于它存在着一定的随机性,所以用于视觉聚类还需要对随机性加以控制或减弱。

参考文献:

[1] Indyk P, Motwani R. Approximate nearest neighbors: towards removing the curse of dimensionality [C] //The Symposium on Theory of Computing. 1998: 604-613.

[2] Gionis A, Indyk P, Motwani R. Similarity search in high dimensions via hashing [C] //The 25th International Conference on Very Large Data Bases. 1999: 518-529.

[3] Datar M, Immorlica N, Indyk P, et al. Locality sensitive hashing scheme based on p-stable distributions [C] //The ACM Symposium on Computational Geometry. 2004: 253-262.

[4] Andoni A, Indyk P. E2lsh: Exact Euclidean locality-sensitive hashing (E²LSH 0.1 User Manual) [EB/OL]. [2005-06-01]. http://www.mit.edu/~andoni/LSH/manual.pdf, October 20, 2011.

[5] Andoni A, Indyk P. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions [J]. Communications

of the ACM, 2008, 51(1): 117-122.

[6] Panigrahy R. Entropy-based nearest neighbor algorithm high dimensions [C] //The ACM-SIAM Symposium on Discrete Algorithms. 2006: 1185-1195.

[7] Buhler J, Tompa M. Finding motifs using random projections [J]. Journal of Computational Biology, 2002, 9(2): 225-242.

[8] Shakhnarovich G, Viola P, Darrell T. Fast pose estimation with parameter-sensitive hashing [C] //The 9th IEEE International Conference on Computer Vision. 2003: 13-16.

[9] Casey M, Slaney M. Fast recognition of remixed music audio [C] //The IEEE International Conference on Acoustics, Speech, and Signal Processing. 2007: 1425-1428.

[10] Liang Ying Yu, Li Jian Min, Zhang Bo. Vocabulary-based Hashing for Image Search [C] //The International Conference on Multimedia. 2009: 589-592.

[11] Jegou H, Douze M, Schmid C. Improving bag-of-features for large scale image search [J]. International Journal of Computer Vision, 2010, 87(3): 316-336.

[12] Yang C. Macs: Music audio characteristic sequence indexing for similarity retrieval [C] //Tthe Workshop on Applications of Signal Processing to Audio and Acoustics. 2001: 123-126.

[13] Liu Zhu, Liu Tao, David G. Effective and scalable video copy detection [C] //The ACM SIGMM International Conference on Multimedia Information Retrieval. 2010: 119-128.

[14] Ficichella M, Deng Feng, Nejdi W. Efficient Near-Duplicate Detection based on Locality Sensitive Hashing [C] //The 21st International Conference on Database and Expert Systems Applications. 2010: 152-166.

[15] Ravichandran D, Pantel P, Hovy E. Randomized Algorithms and NLP: Using Locality Sensitive Hash Function for High Speed Noun Clustering [C] //The 43rd Annual Meeting on Association for Computational Linguistics. 2005: 622-629.

[16] Motwani R, Naor A, Panigrahy R. Lower bounds on locality sensitive hashing [C] //The ACM Symposium on Computational Geometry. 2006: 253-262.

[17] Andoni A. Nearest Neighbor Search: the old, the new, and the Impossible [D]. Doctor Thesis, Massachusetts Institute Technology, USA, 2010.

[18] 高毫林,彭天强,李弼程,等. 基于多表频繁项投票和桶映射链的快速检索方法 [J]. 电子信息学报,2012,34(11): 2574-2581.

[19] 唐俊华,阎保平. 基于 LSH 索引的快速图像检索 [J]. 计算机工程与应用,2002,38(24): 20-22.

[20] 卢炎生,饶祺. 一种 LSH 索引的自动参数调整方法 [J]. 华中科技大学学报,2006,34(11): 38-41.

[21] 蔡衡,李舟军,孙健,等. 基于 LSH 的中文文本快速检索 [J]. 计算机科学,2009,36(8): 201-205.

[22] 刘守群,朱明,郑烃. 一种基于内容相似的重复视频片断检测方法 [J]. 中国科技大学学报,2010,40(11): 1130-1135.

[23] 杨恒,王庆,何周灿. 面向高维图像特征匹配的多次随机子向量量化哈希算法 [J]. 计算机辅助设计与图形图像学报,2010,22(3): 494-503.

[24] 龙柏,孙广中,熊焰,等. 一种基于多核机群架构的混合索引结构 [J]. 电子学报,2011,39(2): 275-279.

[25] Marée R, Denis P, Wehenkel L. Incremental Indexing and distributed Image Search using Shared Randomized Vocabularies [C] //The ACM SIGMM International Conference on Multimedia Information Retrieval Philadelphia. 2010: 29-38.

[26] Lv Q, Josephson W, Wang M. Multi-probe LSH: Efficient indexing for high-dimensional similarity search [C] //The 24rd International Conference on Very Large Data Bases. 2007: 950-961.

[27] Bawa M, Condie T, Ganesan P. Lsh forest: self-tuning indexes for similarity search [C] //The 14th international conference on World Wide Web. 2005: 651-660.

[28] Dong Wei, Wang Zhe, Josephson W, et al. Modeling LSH for Performance Tuning [C] //The association of Computing Machinery Conference on Information and Knowledge Management. 2008: 669-678.

[29] Kulis, B, Grauman K. Kernelized locality-sensitive hashing for scalable image search [C] //The IEEE International Conference on Computer Vision. 2009: 2130-2137.

[30] Kulis B, Jain P, Grauman K. Fast similarity search for learned metrics [J]. IEEE Trans. on PAMI, 2009, 31(12): 2143-2157

[31] 杨丹,李博,赵红. 鲁棒觉词汇本得自适应构造与自然场景分类 [J]. 电子与信息学报, 2010, 32(9): 2139-2144.

[32] 刘硕研,须德,冯松鹤. 一种基于上下文语义信息的图像块视觉单词生成算法 [J]. 电子学报, 2010, 38(5): 1156-1161.

[33] 李华北,胡卫明,罗冠. 基于语义匹配的交互式视频检索框架 [J]. 自动化学报, 2008, 34(10): 1243-1249.

[34] Deng Jia, Alexander C Berg, Li Fei Fei. Hierchical Semantic Indexing for large Scale Image Retrieval [C] //The Computer Vision and Pattern Recognition. 2011:785-792.

[35] Yang Li, Jin Rong, Sukthankar R. Unifying Discriminative Visual Codebook Genaration with Classifier Training for Object Category Recognition [C] //The IEEE Conference on Computer Vision and Pattern Recognition. 2008:11-18.

[36] Lampert C H, Blaschko M B, Hofmann T. Beyond sliding windows: Object Localization by Efficient Subwindow Search [C] //The IEEE Conference on Computer Vision and Pattern Recognition. 2008:1-8.

[37] He Jun Feng, Radhakrishnan R, Chang S F. Compact Hashing with Joint Optimization of Search Accuracy and Time [C] // The IEEE Conference on Computer Vision and Pattern Recognition. 2011:753-759.

[38] Liu Wei, Wang Jun, Kumar S, et al. Hashing with Graphs [C] //The 28th International Conference on Machine Learning. 2011:522-529.

[39] 沈筱彦,陈俊亮,孟祥武,等. 可并行中文同主题词聚类新算法 [J]. 北京邮电大学学报,2009,32(4):121-126.

[40] Mu Ya Dong, Sun Ju, Han Tony X. Randomized Locality Sensitive Vocabularies for Bag-of-Features Model [C] //The 11th European conference on computer vision conference on Computer vision. 2010:748-751.

[41] 吴磊. 视觉语义分析: 从底层特征表达到语义距离学习 [D]. 合肥: 中国科学技术大学,2010.

[42] Moosmann F, Nowak E, Jurie F. Randomized clustering forests for image classification [J]. IEEE Trans. Pattern Anal. Mach Intell, 2008, 30(9):1632-1646.

(上接第 324 页)

时,从图 7 可以看出,在混合信号幅度估计的精度上,本文算法要优于 *max-min* 算法,这是由于本文算法不仅利用了混合部位的数据,而且还通过未混合部位的单个信号来提高估计精度。同时,随着信噪比的增加,本文算法与混合矩阵已知条件下分离性能越来越接近,说明了本文算法的有效性。

4 结束语

本文研究了多通道恒模调制混合信号盲分离问题,将混合矩阵的估计转化为混合信号的幅度估计问题进行求解,降低了复杂度,提高了盲分离性能。在进行幅度估计时,首先基于信号混合部位幅度最大最小值,对混合信号的混合位置进行了估计,将信号划分为混合部位和未混合部位后,利用混合部位的最大最小值和未混合部位的单信号幅度值对混合信号的幅度进行联合估计,提高了估计精度。仿真表明,本文算法接近混合矩阵完美估计下的盲分离性能,具有较强的实用性。

参考文献:

[1] Paolo Burzigotti, Alberto Ginesi, Giulio Colavolpe. Advanced receiver design for satellite-based automatic identification system signal detection [J]. International Journal of Satellite Communications and Networking, 2012, 30:52-63.

[2] Lee W. Overview of Cellular CDMA [J]. IEEE Trans. on VT, 1991, 40:291-301.

[3] 芮国胜,徐彬,张嵩. 同频重叠信号的单通道盲分离算法综述 [J]. 电光与控制,2011,18(9):58-63.

[4] 廖灿辉,涂世龙,万坚. 基于迭代的同频混合信号单通道盲分离/译码算法 [J]. 通信学报,2011,32(8):111-117.

[5] 艾朝霞,刘卫波. 通信信号盲源分离的高效算法研究 [J]. 陕西科技大学学报,2011,29(4):77-81.

[6] 刘伟群,阳春华,易叶青. JADE 盲分离算法失效问题研究 [J]. 计算机技术与自动化,2005,24(3):63-65.

[7] 张华,冯大政,聂卫科,等. 非正交联合对角化盲源分离算法 [J]. 西安电子科技大学学报,2008,35(1):27-31.

[8] 芮国胜,徐彬,张嵩. 单通道混合信号的幅度估计算法 [J]. 通信学报,2011,32(12):82-87.

[9] 徐彬,芮国胜,陈必然. 一种单天线同频混合信号幅度的估计算法 [J]. 电讯技术,2011,51(10):20-23.

[10] Cardoso J F, Souloumiac A. Blind Beamforming for Non-Gaussian Signals [J]. IEE Proceedings-F, 1993, 140(6):362-370.

[11] Comon P. Independent Component Analysis, A New Concept [J]. Signal Processing, 1994, (36):287-314.

[12] 万坚,涂世龙,廖灿辉,等. 通信混合信号盲分离理论与技术 [M]. 北京: 国防工业出版社,2012: 88-90.