大规模图像特征检索中查询结果的自适应过滤

艾列富[1),2)]，于俊清[1+)]，管涛[1)]，何云峰[1)]

[1)](华中科技大学 计算机科学与技术学院，武汉 中国 430074)

[2)]（安庆师范学院 计算机与信息学院 安庆 中国 246133）

摘 要 针对大规模图像的快速检索问题，提出了面向倒排索引结构的检索方法中查询结果的自适应过滤方法：全面过滤和不完全过滤。目的是在不影响查询精度的前提下，提高查询效率。根据查询特征所在的空间位置，全面过滤通过构造以查询特征点为球心的超球体并自适应地计算半径，只对位于超球体内部的查询结果进行排序，从而减少需要排序的查询结果数量，提高查询效率。在此基础上，为了降低过滤查询结果的时间开销，不完全过滤将倒排列表划分为若干子倒排列表并将对应的聚类中心用于过滤查询结果。为了验证所提出方法的有效性，以一种典型检索方法：基于残差量化的检索方法为应用实例，分别将全面过滤和不完全过滤与该检索方法相结合。此外，为了提高特征量化效率，将一种欧式距离下限定理与残差量化相结合并用于过滤特征量化过程中非近邻聚类中心。通过在公开数据集进行实验，实验结果表明在保证具有相同平均查全率的前提下，全面过滤和不完全过滤都能明显减少基于残差量化的检索方法的查询时间，不完全过滤比全面过滤具有更快的检索速度。此外，非近邻聚类中心过滤可以有效提高残差量化的特征量化效率。

关键词 大规模图像特征；查询结果；自适应过滤；超球体；距离下限

Adaptively Filtering Query Results for Large Scale Image Feature Retrieval

AI Lie-Fu[1),2)], YU Jun-Qing[1+)], GUAN Tao[1)], HE Yun-Feng[1)]

1)(School of Computer Science&Technology, Huazhong University of Science and Technology, Wuhan 430074, China)

2)(School of Computer & Information, Anqing Normal University, Anqing 246133, China)

Abstract Aiming at the problem of rapid retrieval in large scale image, two methods: exhaustive filtration and non-exhaustive filtration are proposed to filter query results for inverted indexing structure-based retrieval methods. The objective is to improve query efficiency without influencing accuracy. Exhaustive filtration constructs a hyper-sphere whose center is the query feature, and the corresponding radius is calculated adaptively. Only the features that lie in the hyper-sphere are used to sort, then the number of features need to be sorted is reduced and the query efficiency is increased. Based on this, to reduce the time costs on filtering query results, non-exhaustive filtration partitions the inverted list into several sub-inverted lists, where the corresponding centroids are used to filter query results. To demonstrate the effectiveness of proposed methods, a typical method: residual vector quantization-based (RVQ) retrieval is used as an application example, which is combined with exhaustive filtration and non-exhaustive filtration respectively. Besides, to improve the efficiency on quantizing feature vectors, RVQ is combined with a theorem of lower bound of Euclidean distance which is used to filter non-nearest centroids in vector quantization process. The experimental results on public datasets show that both exhaustive filtration and non-exhaustive can noticeably reduce the query time of RVQ-based retrieval in the condition of same average recall rate. Moreover, non-exhaustive filtration is faster than exhaustive filtration. Besides, RVQ can be efficiently improved by filtering non-nearest centroids.

---

*本课题得到国家自然科学基金（61173114、61202300和61272202）和武汉市应用基础研究计划项目（2014010101010027).

艾列富，男，1985年生，博士，主要研究领域为基于内容的大规模图像高维索引与检索，E-mail: ailiefuhu@gmail.com. 于俊清（通讯作者），男，1975年生，博士，教授，主要研究领域为为数字媒体处理与检索，多核处理器编程环境，E-mail: yjqing@hust.edu.cn. 管涛，男，1978年生，博士，副教授，主要研究领域为移动视觉搜索，增强现实，计算机视觉，E-mail: qd_gt@126.com. 何云峰，男，1977年生，博士，讲师，主要研究领域为数字媒体处理与检索，E-mail: yfhe@hust.edu.cn.

Key words large scale image feature; query results; adaptive filtration; hyper-sphere; lower bound of distance

1 引言

随着互联网和多媒体技术的发展，以图像为代表的多媒体信息呈现爆炸性增长。面对海量的图像库，只有对图像进行有效地组织以便于浏览、搜索和检索，人们才能快速和准确地获取视觉上相似的图片。因此，如何在保证具有较好查询精度的情况下快速检索到相似图像至关重要。

类似于 kd 树[1,2]的树形检索方法虽然在低维特征空间上具有较好的检索效率，但是随着维度的增加，其搜索效率就会不断降低，最终退化到复杂度为 $o(nd)$ 的线性检索。

类似于 E2LSH[3]的位置敏感哈希方法通过一组哈希函数，将相似的图像特征映射到哈希表中相同桶或者邻近桶。迄今为止，E2LSH 已成功应用于局部描述符[4]和三维物体[5]的索引与检索。相对于稀疏特征向量，E2LSH 对致密特征向量具有更好的检索性能。然而，E2LSH 需要计算查询特征与查询结果集中所有特征之间的欧式距离并排序，这意味着图像的原始视觉特征需要存储在计算机内存中，从而在一定程度上影响了 E2LSH 的检索速度和可以处理的图像库规模。为了降低存储空间,哈希编码方法，如：谱哈希（Spectral Hasing）[6]、球形哈希（Spherical Hashing）[7]、k-means 哈希（k-means Hashing）[8]、随机最大边缘哈希（Random Maximum Margin Hashing）[9]以及迭代量化（Iterative Quantization）[10]等，利用哈希函数将相似的特征映射为相同或者汉明距离相近的二进制编码。这类方法同样需要对所有查询结果进行排序，虽然其使用的汉明距离较欧式距离可以大幅提高排序速度。但是，汉明距离的区分能力受限于其采用的二进制编码的长度。

从文本检索引入视觉检索，基于倒排索引的图像检索方法[11]近年来得到了广泛的研究与应用。倒排索引结构中每个倒排列表由一个视觉单词指示。每个索引列表相当于一个聚类，对应的聚类中心即为视觉单词。构建倒排索引时，图像的视觉特征被插入到距离最近的视觉单词对应的倒排列表。为了降低倒排索引结构所需的存储空间，目前已有一些有损压缩方法以及特征量化方法用于对图像特征进行编码，从而大幅降低图像特征的存储需求。这些方法主要包括汉明嵌入[12,13]、非对称汉明嵌入[14]、miniBOF[15]、积量化[16,17]、残差量化[18]以及转换编码[19]等。传统倒排索引结构中倒排列表数即为需要训练的聚类中心数。倒排列表的规模越大，对数据集的划分粒度则越细，但是训练聚类中心所花费的时间开销就越大。文献[20]利用积量化将特征向量分为 2 段并在子向量空间上分别训练 k 个聚类中心，从而构建包含 k^2 个索引列表的倒排索引结构。倒排列表对应的聚类重心为这 2 组聚类重心的串联向量。图像特征检索时，基于倒排索引的图像检索方法首先查找若干个近邻倒排列表，然后将对应倒排列表中所有特征点都作为查询结果并排序。

目前面向倒排索引的检索方法通常将距离查询特征最近若干个视觉单词对应的倒排列表中所有特征都作为查询结果并用于排序。因而，查询结果的数量是影响查询速度的一个重要因素。实际上，与查询特征相似的仅仅是空间位置位于查询特征周围的特征点。如果能够对查询结果进行有效地过滤，只将位于查询特征周围位置的查询结果用于排序，将对提升查询速度具有重要意义。

本文提出面向倒排索引结构的两种基于自适应超球体的查询结果过滤方法：全面过滤和不完全过滤。全面过滤方法为每个查询特征点自适应地构造以其为球心的超球体，只对查询结果中位于超球体内部的特征点进行排序，从而减少排序的特征点数，提高查询速度。在此基础上，不完全过滤方法通过降低过滤查询结果的计算量，进一步提高查询效率。以基于残差量化的检索方法为应用实例，将提出的自适应过滤方法应用于该检索方法并通过实验证明查询结果的全面过滤和不完全过滤在不影响查询精度的前提下，可以明显减少查询时间。此外，目前的量化方法，如积量化[16,17]、残差量化[18]等，都是采用硬匹配方法，即计算输入图像特征与量化器每个聚类中心的距离并找到距离最近的聚类中心作为该层的量化结果。因而，随着聚类中心的数量或者图像特征规模增大，其特征量化的时间开销呈线性增长。文献[21]提出了一种基于下限过滤的精确最近邻查找方法用于图像分类。本文将利用文献[21]提出的欧式距离下限同残差量化方法相结合以提高特征量化的效率

本文第二节介绍相关工作；第三节分别论述提出的两种查询结果自适应过滤方法及其在基于残差量化的检索方法中应用；第四节描述基于下限过滤的特征量化；第五节是实验性能评测和比较；第

六节对全文进行总结和讨论。

2 相关工作

传统基于倒排索引的图像检索方法通常在倒排索引结构中查找距离查询图像的视觉特征最近 w（w≥1）个视觉单词[16,18]，并将这些视觉单词对应的倒排列表中所有视觉特征都作为查询结果。如果最终查询结果只需要 knn 个相似的视觉特征，则需要进一步计算查询特征与所有查询结果之间的欧式距离并排序。

基于残差量化的检索方法[18]是在传统面向倒排索引的检索方法的基础上提出的一种基于量化的搜索方法。构建倒排索引结构时，首先类似于传统方法，根据用 k-means 训练得到的视觉单词本 $\upsilon=\{c_1,c_{1,},...,c_k\}$，将特征 x_i 插入到距离最近的视觉单词对应的倒排列表，其计算方法如公式（1）所示：

$$c_x=\arg\min_{c_j\in\upsilon,\ j=1,..,k} d(x_i,c_j) \qquad (1)$$

其中，c_x 为距离 x_i 最近的视觉单词，$d(x_i,c_j)$ 为 x_i 与索引结构中各视觉单词之间的欧氏距离。然后，将利用残差量化方法对 x_i 进行量化得的编码保存对应倒排列表。图 1 为对应的倒排索引示意图，每个倒排列表对应一个视觉单词，相当一个聚类，倒排列表的每个节点存储视觉特征 ID 及其残差量化编码。其将 x_i 插入倒排索引的时间复杂度为 O(k×dim)，其中 dim 为特征的向量维度。

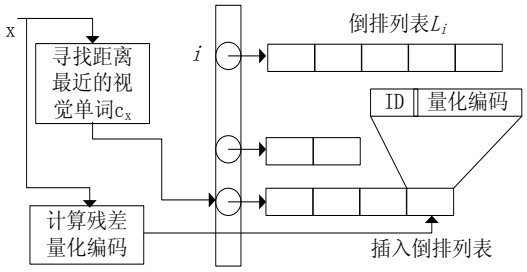

图 1 基于残差量化的倒排索引结构示意图

类似于传统检索方法，基于残差量化的图像检索方法首先查找距离查询特征最近的 w 个视觉单词；然后将这 w 个视觉单词对应倒排列表中所有特征都作为初始查询结果；最后，利用查询特征与所有查询结果之间的非对称距离来对查询结果进行排序，最终得到 knn 个最相似的查询结果。

3 基于自适应超球体的查询结果过滤

为了只对查询结果中空间位置位于查询特征点周围的特征点排序以减少排序特征数量，提出两种基于自适应超球体的查询结果过滤方法：全面过滤和不完全过滤。为了便于下一节通过实验证明所提出的方法在不影响查询精度的前提下，可有效提高面向倒排索引的检索方法的查询速度，本节以基于残差量化的检索方法为应用实例，对查询结果的自适应过滤方法进行论述。

3.1 全面过滤

对于查询图像特征，在检索最近的 w 个倒排列表后，查询结果的全面过滤就用于对这 w 个倒排列表中的特征点进行过滤，得到位于查询特征点周围位置的特征点并用于排序。其关键在于确定一个适当的半径，在特征空间构造一个以查询特征点为球心的超球体，过滤掉查询结果中位于超球体之外的特征点，只对超球体内部的查询结果进行排序。

给定一个倒排索引结构 $L=\{l_1,l_2,...,l_k\}$，对应的聚类中心为 $C=\{c_1,c_2,...c_k\}$。对于查询特征 q，全面过滤应用于基于残差量化的检索方法的具体查询过程如下：

Step1: 分别计算 q 与 C 中所有聚类中心的欧式距离 $q=\{d_1,d_2,...d_k\}$；

Step2: 从 d 中取最小的前 w 个距离值 $\{d_{q,1},d_{q,2},...d_{q,w}\}$，并将对应倒排列表中所有特征点都作为查询结果 $RS_q=\{y_1,y_2,...y_m\}$；

Step3: 在特征空间中构建以 q 为球心的超球体，其中，超球体半径 $Radius_q$ 根据 q 到最近的 w 个聚类中心的距离 $d_{q,i}$ 利用以下公式计算：

$$Radius_q=\lambda\times\frac{1}{w}\sum_{i=1}^{w} d_{q,i} \qquad (2)$$

其中，λ 为比例系数，用于调整半径的大小，目的是获得不影响查询精度的最小半径，从而尽可能多地过滤掉不相似查询结果；

Step4: 利用基于残差量化的检索方法中非对称距离计算方法，计算查询特征 q 与 RS_q 中特征之间的距离 $d(q,y_i)$，只保留满足公式（3）的查询结果，得到待排序查询结果 $RS_{qnew}=\{y'_1,y'_2,...y'_b\}$；

$$d(q,y_i)=\|q-y_i\|\le Radius_q \quad (i=1,2,...m) \qquad (3)$$

Step5：根据 RS_{qnew} 中特征与查询特征之间的距

离值对其进行排序，返回距离值最小的 knn 个特征点作为最终的查询结果。

其中，Step3 和 Step4 为查询结果全面过滤的具体步骤，其余步骤与传统基于倒排索引的检索方法和基于残差量化的检索方法相同。全面过滤并没有改变倒排索引结构的构建方式，只是在特征查询过程中，增加了查询结果过滤的机制。因此，对于倒排索引结构 $L=\{l_1,l_2,...,l_k\}$，应用全面过滤时将图像特征 x 插入索引结构的过程类似于图 1，其时间复杂度依然为 O(k×dim)。

3.2 不完全过滤

全面过滤方法通过在特征空间中为查询特征构造一个超球体，将位于超球体之外的查询结果特征点过滤掉，从而减少排序的查询结果数量。然而，全面过滤需要计算查询特征与查询结果中所有特征的距离并同对应的超球体半径进行比较。为了进一步提高查询速度，不完全过滤在查询结果过滤过程中减少查询结果特征与查询特征之间的距离计算次数及其同超球体半径的比较次数。

为此，不完全过滤通过将倒排索引结构中倒排列表划分为若干个子倒排列表并将对应的聚类中心用于过滤查询结果。该过程是在构建倒排索引时通过训练所需聚类中心来完成的。

以图 1 所示的基于残差量化的检索方法的索引结构为例，在构建倒排索引的训练阶段，利用层次 k-means 方法（Hierarchical k-means，HKM）[22]在一个特征样本集上训练两层聚类中心：C 和 $\{C_{ci}\}$。其中，第一层包含 k_1 个聚类中心 $C=\{c_1,...,c_i,...c_{k1}\}$；$c_i\,(i=1,2,...,k_1)$ 对应的聚类被再次划分为 k_2 类，对应的第二层聚类中心为 $C_{ci}=\{c_{i1},...,c_{ij},...,c_{ik2}\}$。

将图像特征 x 插入索引结构时，首先计算 x 到 C 中各聚类中心的距离并利用公式（1）找到最近的聚类中心 c_i；然后，计算 x 到 c_i 对应的 C_{ci} 中 k_2 个聚类中心的距离，同样利用公式（1）找到最近的聚类中心 c_{ij}；最后将 x 的 ID 及其元数据插入到 c_{ij} 对应的倒排列表。因而，将 x 插入倒排索引结构的时间复杂度为 $O((k_1+k_2)\times dim)$, 其中 dim 为特征的向量维度。如图 4 所示，若将不完全过滤应用于基于残差量化的检索方法时，存入倒排列表的元数据为图像特征的残差量化编码。

对比图 1，图 2 所示的倒排索引结构相当于将图 1 中每个倒排列表 L_i 划分为 k_2 类形成 k_2 个子倒排类表，并且 C_{ci} 中的聚类中心 c_{ij} 用于标识倒排列表 L_{ij}。

对于查询图像特征 q，结合不完全过滤的基于残差量化的检索方法的查询过程如下：

Step1: 分别计算 q 到所有聚类中心 $C=\{c_1,c_2,...c_{k1}\}$ 的距离得到 $D_1=\{d_1,d_2,...d_{k1}\}$；

Step2: 从 D_1 中查找距离最小的 w 个值 $\{d_{q,1},d_{q,2},...d_{q,w}\}$，其对应的聚类中心为 $C_{cq}=\{c_{q,1},c_{q,2},...,c_{q,w}\}$；

Step3: 将与这 w 个聚类中心对应的第二层聚类中心合并形成集合 $C_{cq}'=\bigcup_{i=1}^{w}C_{c_{q,i}}=\bigcup_{i=1}^{w}\bigcup_{j=1}^{k2}c_{c_{q,i}j}$，并将所有 $c_{c_{q,i}j}$ 对应的倒排列表中特征点都作为 q 的初始查询结果 $RS_{qo}=\bigcup_{i=1}^{w}\bigcup_{j=1}^{k2}l_{c_{q,i}j}$；

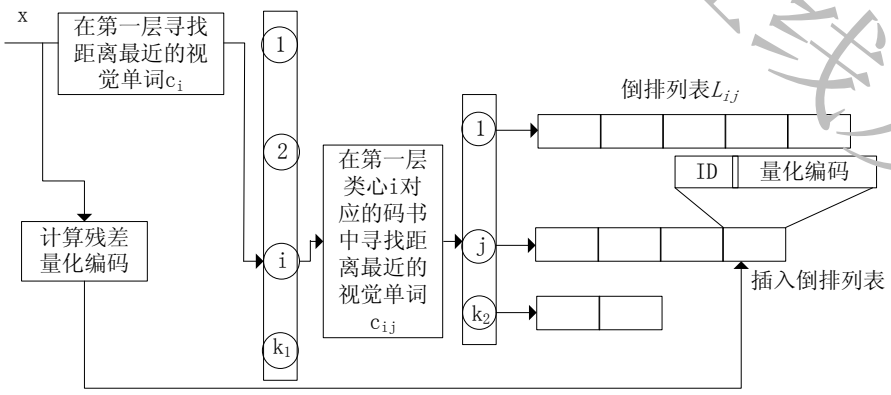

图 2 不完全过滤对应的倒排索引结构示意图

Step4: 同全面过滤，在特征空间中构造以 q 为球心的超球体 S_q，对应的超球体半径 $Radius_q$ 利用公式（2）根据 q 到 C 中最近的 w 个聚类中心的距

离来计算；

Step5: 过滤不相似查询结果：计算查询特征 q 与 C_{cq}' 中所有聚类中心 $c_{c_{q,i\,j}}$ 之间的欧式距离，利用公式（4）过滤不相似查询结果，得到待排序查询结果集合 $RS_q=\bigcup_{i=1}^{w}\bigcup_{j=1}^{k_2} l_{c_{q,i\,j}}$，$(d(q,c_{c_{q,i\,j}})\le \mathrm{R_q})$：

$$d(q,c_{c_{q,i\,j}})=\left\|x-c_{c_{q,i\,j}}\right\|\le Radius_q \quad (i=1,...w,\ j=1,...,k_2) \qquad (4)$$

如果 $c_{c_{q,i\,j}}$ 位于超球体的内部，就认为对应倒排列表 $l_{c_{q,i\,j}}$ 中所有特征点都与 q 相似，反之不相似。由于每个倒排列表 $l_{c_{q,i\,j}}$ 相当于一个聚类，聚类里的所有特征都认为是与聚类中心 $c_{c_{q,i\,j}}$ 相似的，因此可以用 $c_{c_{q,i\,j}}$ 来代表 $l_{c_{q,i\,j}}$ 中特征并用于不相似特征过滤；

Step6: 利用基于残差量化的检索方法中非对称距离方法，根据查询特征 q 与 RS_q 中特征之间的距离进行排序，返回距离最小的 knn 个查询结果。

相比查询结果的全面过滤，不完全过滤通过将每个倒排列表再次划分为 k_2 个聚类并将对应的聚类中心代表对应的子类，用于非相似特征的过滤，从而大幅度降低了过滤非相似特征时距离计算次数。在都检索最近的 w 个聚类中心后，得到初始的查询结果中特征数量为 n 的前提下，如果经全面过滤用于排序的特征点数量为 n' 并且采用快速排序算法对其排序，那么应用全面过滤的检索方法的复杂度为 $\mathrm{O}(n\times\dim+n'\log n')$；如果经不完全过滤用于排序的特征点数量为 n''，同样采用快速排序对其排序，那么应用不完全过滤的检索方法的复杂度为 $\mathrm{O}((w\times k_2+n'')\times\dim+n''\log n'')$，dim 为特征维度。

4 基于下线过滤的特征量化

利用文献[21]提出的基于下限过滤的非近邻查找方法，本文以残差量化为例，将其与残差量化相结合，以提高图像特征量化和编码的效率。

文献[21]提出了一种计算特征向量之间欧式距离下限的定理：

$$d(x,y)^2\ge \dim\times((\mu_x-\mu_y)^2+(\sigma_x-\sigma_y)^2)=lb(x,y) \qquad (5)$$

其中，dim 为图像特征向量的维度，μ_x、μ_y、σ_x 和 σ_y 分别是 x 和 y 中向量分量的均值和标准差。$(\mu_x-\mu_y)^2+(\sigma_x-\sigma_y)^2$ 即为两个二维向量（μ_x，σ_x）和（μ_y，σ_y）的欧式距离平方。

通过将公式（5）与残差量化相结合，在一个二维空间计算待量化特征与聚类中心之间欧式距离的下限并过于过滤非近邻聚类中心和寻找距离最近聚类中心，进而提高特征量化的效率。其量化图像特征的具体过程如算法 1 所示：

算法 1. 结合距离下限过滤的残差量化方法.

输入：图像特征 x, μ_x, σ_x；RVQ 码书：$\{C_l\}=\{\{c_{li}\}\},\{\mu_l\}=\{\{\mu_{c_l}\}\},\{\sigma_l\}=\{\{\sigma_{c_{li}}\}\}$ i=1...k,l=1...L

输出：量化结果 $\{c_1^{\,nearest},...,c_L^{\,nearest}\}$

```
FOR l = 1 → L
  从 C_l 随机选取一个聚类中心 c' 作为种子 c_l^nearest
  计算欧式距离 min_d ← d(x,c_l^nearest)^2
  FOR i = 1 → k
    IF lb(x,c_li) ≥ min_d THEN
      CONTINUE
    ELSE
      计算 d(x,c_li)^2
      IF d(x,c_li)^2 ≥ min_d THEN
        CONTINUE
      ELSE
        c_l^nearest ← c_li
        min_d ← d(x,c_li)^2
      END IF
    END IF
  END FOR
  x ← x − c_l^nearest
END FOR
RETURN {c_1^nearest,...,c_L^nearest}
```

同残差量化，图像特征 x 按照从第 1 层到第 L 层的顺序进行量化，每层量化的输入为上一层量化误差。在每一层的特征量化过程中，首先从 k 个聚类中心中随机选择 c' 作为的初始最近邻聚类中心

（种子点）并将距离 $d(x,c_l^{nearest})^2$ 作为对应的最小距离 min_d (第 2-3 行)。$c_l^{nearest}$ 用于记录特征向量 x 的当前最新精确最近邻聚类重心（第 4-17 行）。当 $lb(x,c_{li}) \geq \min\_d$ 时，使用距离下限可以有效过滤聚类中心 c_{li}。当 x 与 c_{li} 之间的距离下限满足 $lb(x,c_{li}) \geq \min\_d$ 时，就说明 x 到 c_{li} 的的欧式距离 $d(x,c_{li})^2$ 一定满足 $d(x,c_{li})^2 \geq \min\_d$，因此 c_{li} 肯定不是 x 的最近邻聚类中心而被过滤掉。反之，则需要进一步计算 $d(x,c_{li})^2$ 并与 min_d 进行比较。

相比在原始高维特征空间上计算特征向量 x 与聚类中心 c_{li} 之间的距离 $d(x,c_{li})^2$，在一个二维特征空间上计算两个 2 维向量之间的距离会大幅降低计算量。因此，算法 1 可以明显提高残差量化的量化效率，尤其当高维图像特征以及大规模图像特征，其提升效果将更为明显。相比硬匹配方法需要计算图像特征 x 到所有聚类中心的距离，其复杂度是线性依赖于聚类重心的数量，而算法 1 通过欧式距离下限过滤非近邻聚类中心，只需要计算 x 到部分聚类中心的距离 $d(x,c_{li})^2$。因此，结合距离下限过滤的残差量化方法的量化效率是非线性依赖于聚类中心数量的。

5 实验

5.1 实验数据集和实验环境

本文将在公开的 sift 特征数据集[16]和 gist 数据集上测试和评估全面过滤和不完全过滤的近邻搜索性能。数据集的具体信息如表 1 所示：

表 1 数据集信息

数据集	sift	gist
特征维度	128	960
训练集规模	100,000	500,000
数据库规模	1,000,000	1,000,000
查询集规模	10,000	1,000

所有实验都是在一台 Intel Core i5 2.8GHZ CPU, 4G 内存的 PC，MATLAB 2011 环境下完成的。

5.2 基于残差量化的检索性能

表 2 给出了 sift 和 gist 数据集上基于残差量化的检索方法对于参数 $k \in \{64, 256, 1024\}$ 和 w 值获得的最高平均查全率（Recall@R）及查询速度。其中，R 为最终排序返回的查询结果数，设为 100；k 与 w 分别为索引结构中倒排列表数和查找的倒排列表数。后面的实验将在表 2 所示的参数设置（k，w）的基础上，对全面过滤和不完全过滤应用于基于残差量化的检索方法时得到的查询性能进行评估并将其与表 2 所示的实验结果比较，目的是证明所提出的基于自适应超球体的查询结果过滤方法可以为面向倒排索引的检索方法有效地降低待排序特征点数和提高查询速度。

表 2 基于残差量化的检索在各种索引结构下的检索性能

sift	Recall@100	查询时间（毫秒）	排序的特征点数
k=64, w=8	0.94	21.8	140280
k=256, w=16	0.94	11.8	66612
k=1024,w=32	0.95	6.9	34509
gist			
k=64, w=8	0.66	27.1	163139
k=256, w=16	0.66	15.9	84818

5.3 评估最优的比例系数 λ

5.3.1 全面过滤中 λ

根据表 2 提供的参数（k，w），表 3 和表 4 分别是应用了全面过滤的基于残差量化的检索方法在 sift 和 gist 数据集上对于不同比例系数 λ 的查询性能，其中 k 与 w 分别为索引结构中倒排列表对应的聚类中心总数和查找的聚类中心数。λ 用于调整超球体的范围，目的是获得一个最优 λ 值，从而构造最小的超球体，在保持查询精度不变的前提下，最大程度上降低排序的特征点数量和提高查询速度。查询特征 x 与聚类中心 c_i $(i=1,2,\ldots k)$ 的平方距离 $d(x,c_i)^2$ 如公式（6）所示。

$$d(x,c_i)^2 = \|x-c_i\|^2 = \|x\|^2 + \|c_i\|^2 - 2 < x, c_i > \quad (6)$$

其中，$\|x\|^2$ 和 $\|c_i\|^2$ 分别是特征向量 x 和聚类中心 c_i 的模的平方，$< x, c_i >$ 为 x 和 c_i 的向量内积。

计算 x 到 c_i 距离的最终目的是为了得到距离最近的 w 个聚类中心，而计算 x 到所有聚类中心的距

离都要计算$\|x\|^2$，因而，是否计算其模值$\|x\|^2$并不影响找到最近的 w 个聚类中心，因此，为了降低计算量，利用$D(x,c_i)$来代替 x 到c_i的平方距离$d(x,c_i)^2$，$D(x,c_i)$的计算方式如公式（7）：

$$D(x,c_i)=d(x,c_i)^2-\|x\|^2=\|c_i\|^2-2<x,c_i> \quad (7)$$

当在 sift 和 gist 数据集上用程序实现全面过滤方法时，发现$D(x,c_i)$均为负数。因此，用$D(x,c_i)$替换公式（2）中$d_{q,i}$时，对应的超球体半径 Radius_q 就变成了负数。因而，λ越大则超球体的半径越小，反之，超球体的半径就越大。Radius_q主要是作为一个阈值用于过滤查询结果中非相似特征。为了统一，q 到所有查询结果之间的距离$d(q,y)$同样用$D(x,y)$来代替并利用公式（7）计算。

从表 3 和表 4 可以看出，对于不同规模的索引结构，当$\lambda=1$时，应用了全面过滤的基于残差量化的检索方法都能在 Recall@R 与原始方法相同的情况下获得最快的查询速度，因而λ的最优取值为 1。此外还说明，对于全面过滤来说，超球体半径中比例系数λ的取值不依赖于 k 和 w 以及数据集类型。

表 3　不同 λ 时全面过滤在 sift 数据集上的查询性能

sift	λ	Recall@100	查询时间（毫秒）	排序的特征点数
k=64, w=8	0.98	0.94	15.3	9783
k=64, w=8	**1**	**0.94**	**14.8**	**7852**
k=64, w=8	1.1	0.92	13.6	2494
k=256, w=16	0.98	0.94	9.3	5455
k=256, w=16	**1**	**0.94**	**8.7**	**4160**
k=256, w=16	1.1	0.89	8.2	991
k=1024,w=32	0.98	0.95	5.8	3125
k=1024,w=32	**1**	**0.95**	**5.6**	**2315**
k=1024,w=32	1.1	0.86	5.1	457

表 4　不同 λ 时全面过滤在 gist 数据集上的查询性能

gist	λ	Recall@100	查询时间（毫秒）	排序的特征点数
k=64, w=8	0.98	0.66	18.5	15188
k=64, w=8	**1**	**0.66**	**18.0**	**8868**
k=64, w=8	1.1	0.35	16.4	2541
k=256, w=16	0.98	0.66	11.6	8340
k=256, w=16	**1**	**0.66**	**10.9**	**4561**
k=256, w=16	1.1	0.31	10.6	1255

5.3.2　不完全过滤中λ

表 5 和表 6 分别给出了 sift 和 gist 数据集上应用了不完全过滤的基于残差量化的检索方法在不同参数(k_1,w,k_2)下对于不同比例系数λ的查询性能。其中，k_1和 w 取值与全面过滤中 k 和 w 的取值相同，$k_2\in\{4,16,32,48,64\}$。同样采用全面过滤中的方式计算距离和超球体半径，因而，λ越大则超球体半径越小，反之半径越大。

表 5　sift 数据集上不完全过滤对于不同 λ 的查询性能

sift	k_2	λ	Recall@100	查询时间（毫秒）	排序特征点数
k1=64, w=8	4	0.97	0.94	10.8	70486
k1=64, w=8	**4**	**0.98**	**0.94**	**10.4**	**65474**
k1=64, w=8	4	0.99	0.93	9.6	60492
k1=64, w=8	16	0.97	0.94	8.1	48826
k1=64, w=8	**16**	**0.98**	**0.94**	**7.6**	**44903**
k1=64, w=8	16	0.99	0.93	7.0	41119
k1=64, w=8	32	0.97	0.84	7.7	44471
k1=64, w=8	**32**	**0.98**	**0.94**	**7.1**	**40901**
k1=64, w=8	32	0.99	0.93	6.4	37474
k1=64, w=8	48	0.97	0.94	7.4	43681
k1=64, w=8	**48**	**0.98**	**0.94**	**6.9**	**40239**
k1=64, w=8	48	0.99	0.93	6.5	36952
k1=64, w=8	64	0.97	0.94	6.4	35155
k1=64, w=8	**64**	**0.98**	**0.94**	**5.8**	**32073**
k1=64, w=8	64	0.99	0.93	5.3	29116
k1=256, w=16	4	0.94	0.94	7.1	41651
k1=256, w=16	**4**	**0.95**	**0.94**	**6.6**	**38490**
k1=256, w=16	4	0.96	0.93	6.0	35240
k1=256, w=16	16	0.94	0.94	5.7	31942
k1=256, w=16	**16**	**0.95**	**0.94**	**5.3**	**29282**
k1=256, w=16	16	0.96	0.93	5.0	26651
k1=256, w=16	32	0.94	0.94	5.2	27653
k1=256, w=16	**32**	**0.95**	**0.94**	**4.9**	**25248**
k1=256, w=16	32	0.96	0.93	4.5	22863
k1=256, w=16	48	0.94	0.94	5.1	25846
k1=256, w=16	**48**	**0.95**	**0.94**	**4.9**	**23542**
k1=256, w=16	48	0.96	0.93	4.6	21291
k1=256, w=16	64	0.94	0.94	5.1	23783
k1=256, w=16	**64**	**0.95**	**0.94**	**4.9**	**21616**
k1=256, w=16	64	0.96	0.93	4.6	19522

表 5(续表) sift 数据集上不完全过滤对于不同 λ 的查询性能

sift	k_2	λ	Recall@100	查询时间（毫秒）	排序特征点数
k1=1024,w=32	4	0.95	0.95	4.5	21420
k1=1024,w=32	**4**	**0.96**	**0.95**	**4.2**	**19431**
k1=1024,w=32	4	0.97	0.94	4.0	17400
k1=1024,w=32	16	0.95	0.95	4.0	16041
k1=1024,w=32	**16**	**0.96**	**0.95**	**3.9**	**14420**
k1=1024,w=32	16	0.97	0.94	3.7	12831
k1=1024,w=32	32	0.93	0.95	4.5	16182
k1=1024,w=32	**32**	**0.94**	**0.95**	**4.2**	**14744**
k1=1024,w=32	32	0.95	0.94	4.0	13304
k1=1024,w=32	48	0.92	0.95	4.8	15612
k1=1024,w=32	**48**	**0.93**	**0.95**	**4.4**	**14301**
k1=1024,w=32	48	0.94	0.94	4.2	12980
k1=1024,w=32	64	0.90	0.95	5.2	16526
k1=1024,w=32	**64**	**0.91**	**0.95**	**5.0**	**15305**
k1=1024,w=32	64	0.92	0.94	4.9	14572

表 6　gist 数据集上不完全过滤对于不同 λ 的查询性能

gist	k_2	λ	Recall@100	查询时间（毫秒）	排序特征点数
k1=64, w=8	4	0.98	0.66	16.2	96680
k1=64, w=8	**4**	**0.99**	**0.66**	**14.3**	**79298**
k1=64, w=8	4	1	0.65	11.4	61735
k1=64, w=8	16	0.98	0.66	13.4	77326
k1=64, w=8	**16**	**0.99**	**0.66**	**12.1**	**62499**
k1=64, w=8	16	1	0.65	9.5	48422
k1=64, w=8	32	0.98	0.66	13.4	70189
k1=64, w=8	**32**	**0.99**	**0.66**	**11.1**	**56444**
k1=64, w=8	32	1	0.65	9.2	43562
k1=64, w=8	48	0.98	0.66	12.8	66913
k1=64, w=8	**48**	**0.99**	**0.66**	**10.7**	**53412**
k1=64, w=8	48	1	0.65	9.1	41340
k1=64, w=8	64	0.98	0.66	13.3	64618
k1=64, w=8	**64**	**0.99**	**0.66**	**10.8**	**51678**
k1=64, w=8	64	1	0.65	9.3	39820
k1=256, w=16	4	0.97	0.67	13.0	62376
k1=256, w=16	**4**	**0.98**	**0.67**	**10.2**	**53918**
k1=256, w=16	4	0.99	0.66	8.7	43251
k1=256, w=16	16	0.98	0.67	9.4	44133
k1=256, w=16	**16**	**0.99**	**0.67**	**8.5**	**34805**
k1=256, w=16	16	1	0.66	6.7	25745
k1=256, w=16	32	0.98	0.67	9.5	40637
k1=256, w=16	**32**	**0.99**	**0.67**	**8.3**	**31725**
k1=256, w=16	32	1	0.66	7.1	23461
k1=256, w=16	48	0.99	0.67	8.7	29883
k1=256, w=16	**48**	**1**	**0.67**	**7.7**	**21956**
k1=256, w=16	48	1.1	0.22	4.3	2781
k1=256, w=16	64	0.98	0.66	10.4	37334
k1=256, w=16	**64**	**0.99**	**0.66**	**9.2**	**28760**
k1=256, w=16	64	1	0.65	8.4	20952

从表 5 和表 6 可以看出，对于每组参数（k_1, w, k_2），λ 的不同取值虽然能够影响被过滤掉的非相似查询结果数以及 Recall@R （R 取值 100）和查询时间。但是最终都会得到一个最优 λ 值，使得 Recall@R 与原始方法一致并且查询时间最少。即：当 λ 大于最优值时，其 Recall@R 就会下降，反之，Recall@R 始终会保持不变，而查询时间会由于过滤掉的非相似特征的减少而相应地增加。后面的实验中将采用最优 λ 值得到的实验数据。

5.4　不完全过滤对于不同 k_2 值的查询性能

在 $(k1, w) \in \{(64,8),(256,16),(1024,32)\}$ 和最优 λ 值的情况下，不完全过滤在 sift 数据集上对于 k_2 的不同取值时用于排序的特征点数如图 6 所示，图 7 是与图 6 相对应的查询时间。从图 6 可以看出，当 $k_1 = 64, w = 8$ 和 $k_1 = 256, w = 16$ 时，需要排序的查询结果数随着 k_2 的增大而不断减少。然而，当 $k_1 = 1024, w = 32$ 时，这种现象并没有出现，而是趋近于水平。其原因在于随着 k_1 和 k_2 的不断增大，数据库中位于查询特征点对应超球体内部的特征点数趋于平衡。综合图 3 和图 6，当 $k_1 = 256, w = 16$ 和 $k_1 = 1024, w = 32$ 时，查询时间并没有随着排序特征点数的减少而相应地减少。这是因为随着 k_2 和 w 的不断增大，不完全过滤用于计算查询特征到第二层聚类中心的欧式距离的次数以及判断聚类中心是否位于超球体内部所花费的时间也在相应的增加，从而影响总体的查询时间。因此，可以根据实验结果选择不同规模倒排索引结构下适用于不完全过滤的最优 k_2 值。图 5 和图 6 分别是 gist 数据集上与图 3 和图 4 对应的实验结果并且可以从中得出相同的结论。

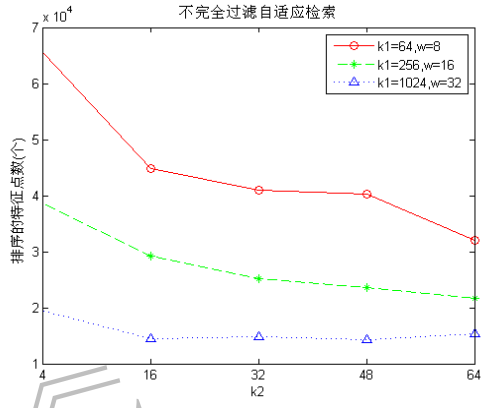

图 3 sift 数据集上不同 k_2 值对应的排序特征点数

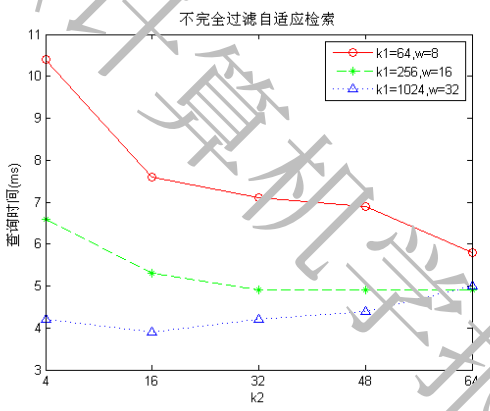

图 4 sift 数据集上不同 k_2 值对应的查询时间

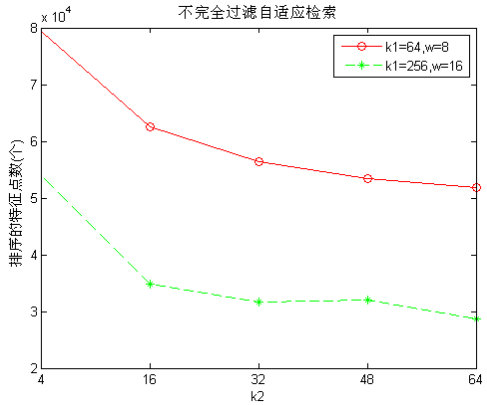

图 5 gist 数据集上不同 k_2 值对应的排序特征点数

5.5 不同检索方法的查询时间对比

简单起见，下面用 RVQ 表示基于残差量化的检索，EF-RVQ 表示应用了全面过滤的基于残差量化的检索，NEF-RVQ 表示应用了不完全过滤的基于残差量化的检索。表 7 和表 8 分别给出了这三种检索方法在不同数据集和不同规模倒排索引结构下的查询性能对比。

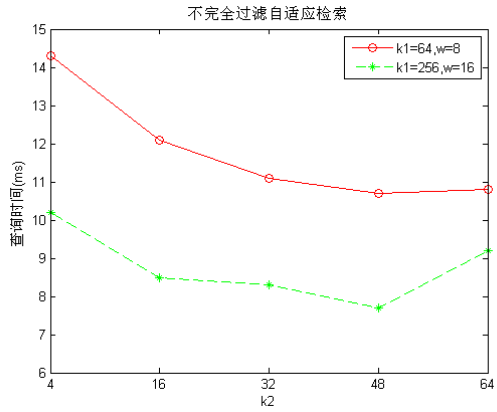

图 6 gist 数据集上不同 k_2 值对应的查询时间

表 7 sift 数据集上 RVQ、EF-RVQ 和 NEF-RVQ 的检索性能对比

sift	Recall@100	查询时间（毫秒）	排序特征点数
RVQ k=64, w=8	0.94	21.8	140280
EF-RVQ k=64, w=8, λ =1	0.94	14.8	7852
NEF-RVQ k_1=64, w=8, λ =0. 98, k_2=64	0.94	5.8	32073
RVQ k=256, w=16	0.94	11.8	66612
EF-RVQ k=256, w=16, λ =1	0.94	8.7	4160
NEF-RVQ k=256, w=16, λ =0. 95, k_2=32	0.94	4.9	25248
RVQ k=1024,w=32	0.95	6.9	34509
EF-RVQ k=1024, w=32, λ =1	0.95	5.6	2315
NEF-RVQ k_1=1024, w=32 λ =0. 96, k_2=16	0.95	3.9	14420

从表 7 和表 8 可以看出，在相同规模的索引结构和获得相同 Recall@100 的情况下，EF-RVQ 的特征查询时间明显少于 RVQ，而 NEF-RVQ 又比 EF-RVQ 进一步减少了查询时间。此外，EF-RVQ 和 NEF-RVQ 较 RVQ 都很大程度上减少了排序的查询结果数。

虽然 EF-RVQ 排序的查询结果数少于 NEF-RVQ，但是 NEF-RVQ 的查询效率要优于 EF-RVQ。其原因在于，EF-RVQ 计算所有查询结果（RVQ 的排序特征点数）到查询特征的距离并与超球体的半径进行比较，而 NEF-RVQ 通过将原始索引列表中特征点划分为 k_2 个聚类并用对应的聚类中心过滤非相似查询结果，从而大幅减少过滤查询

结果所花费的时间，进而较 EF-RVQ 进一步提高了查询效率。

表 8 gist 数据集上 RVQ、EF-RVQ 和 NEF-RVQ 的检索性能对比

gist	Recall@100	查询时间（毫秒）	排序特征点数
RVQ k=64, w=8	0.66	27.1	163139
EF-RVQ k=64, w=8,λ=1	0.66	18.0	8868
NEF-RVQ k_1=64, w=8,λ=0.99,k_2=48	0.66	10.7	53412
RVQ k=256, w=16	0.66	15.9	84818
EF-RVQ k=256, w=16,λ=1	0.66	10.9	4561
NEF-RVQ k_1=256, w=16,λ=1,k_2=48	0.67	7.7	21956

5.6 特征量化效率对比

简单起见，用 RVQ 表示采用传统硬匹配的残差量化方法，用 filtration-RVQ 表示结合了基于下限过滤的非近邻过滤的残差量化方法。实验中，残差量化的的码书层数 L 设置为 8，每层码书的聚类中心数设为 $k \in \{16, 64, 256\}$ 。

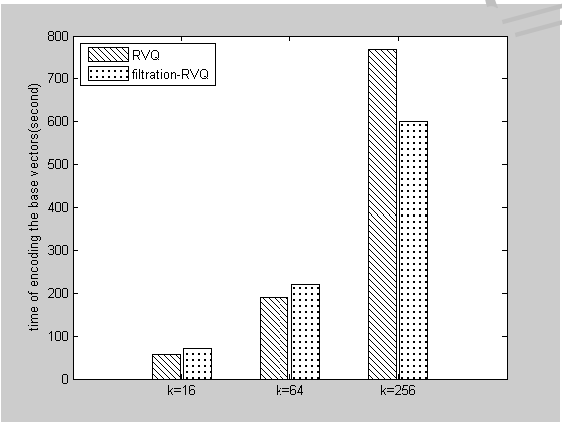

图 7 sift 数据集上量化时间对比

图 7 和 8 分别给出了 RVQ 和 filtration-RVQ 在 sift 和 gist 数据集上量化特征向量的时间对比。从图中可以看出，虽然当 k=16 和 sift 数据集上 k=64 时,filtration-RVQ 的特征量化时间要稍微高于 RVQ，但随着 k 的增大以及特征维度的增加，filtration-RVQ 显示出要比 RVQ 具有更好的时间效率，其表现在量化特征所花费的时间更少并且增长幅度更为平缓。因此，可以得出结论：利用欧式距离下限来过滤非近邻聚类中心，RVQ 的特征量化效率可以得到有效提升，尤其是当 k 比较大以及高维特征时。

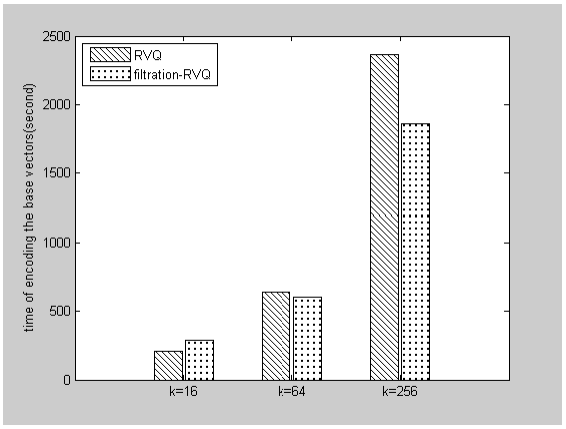

图 8 gist 数据集上量化时间对比

6 总结与讨论

本文提出了两种面向倒排索引的基于自适应超球体的查询结果过滤方法：全面过滤和不完全过滤，并以基于残差量化的检索方法为应用实例来验证所提出方法的有效性。相比基于残差量化的检索方法以及传统检索方法将全部查询结果用于排序，全面过滤为查询特征自适应地计算半径并构造以其为球心的超球体，只对位于超球体内部的查询结果进行排序，从而提高了查询速度。不完全过滤通过将倒排列表划分为若干个子聚类并用这些子聚类的聚类中心过滤非相似查询结果，从而降低了查询结果过滤的时间开销，进一步提高了查询效率。

面向倒排索引的检索方法在索引结构上总体是相似的，如基于残差量化的检索方法和基于积量化的检索方法。它们的倒排索引结构是类似的，不同的是计算查询特征与数据库特征之间距离的计算方式。因而，类似于本文将全面过滤和不完全过滤方法应用于基于残差量化的检索方法的具体过程，这两种查询结果的自适应过滤方法同样可以用类似的方式应用于其他面向倒排索引的检索方法以提高查询效率，如：基于积量化的图像特征检索方法等。

在下一步工作中，我们将在更大的数据集上对提出的两种查询结果过滤方法的有效性进行验证。此外，由于不完全过滤将原始倒排列表划分为若干个子类，从而增加了倒排索引结构的存储需求，因此，未来将在保持查询精度不变的前提下，对平衡查询速度和存储需求作进一步研究。

此外，对于完全过滤，由于查询特征到所有倒排列表对应的聚类中心的距离计算是相互独立的，

不存在相互依赖关系，因此这个过程可以对其进行并行处理，从而进一步提高查询速度；同样的，在过滤非相似特征时，需要计算查询特征与所有查询结果之间的距离，这个过程也可以通过并行处理进行优化。对于不完全过滤，类似于完全过滤，查询特征到第一层索引对应的聚类中心的距离、到第二层用于过滤非相似特征的聚类中心的距离以及到用于排序的查询结果的距离的计算过程同样可以在并行计算环境下进行并行处理，以进一步提高查询速度。这将是下一步的重要研究内容。

参 考 文 献

[1] Beis J S, Lowe D G. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). San Juan, Puerto Rico, 1997:1000-1006

[2] Anan C S, Hartley R. Optimised KD-trees for fast image descriptor matching//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Anchorage, AK, USA, 2008:1-8

[3] Datar M, Immorlica N, Indyk P, Mirrokni V. Locality-sensitive hashing scheme based on p-stable distributions//Proceedings of the 20th Annual Symposium on Computational Geometry. New York, USA, 2004:253-262

[4] Yan K, Rahul S, Larry H. Efficient near-duplicate detection and sub-image retrieval//Proceedings of ACM International Conference on Multimedia. New York, USA, 2004:869-876

[5] Bogdan M, Ying S, Harpreet S S, et al. Rapid object indexing using locality sensitive hashing and joint 3D-signature space estimation. IEEE Transaction on Pattern Analysis and Machine Intelligence, 2006, 28(7):1111-1126

[6] Yair W, Antonio T, Rob F. Spectral hashing. In NIPS. Vancouver, Canada ,2008:1-8

[7] Heo J. P, Lee Y, He J, et al. Spherical hashing//Proceedings of IEEE conference on Computer Vision and Pattern Recognition (CVPR). Providence, RI, USA, 2012:2957-2964

[8] He K, Wen F, Sun J. K. K-means hashing: an affinity-preserving quantization method for learning binary compact codes//Proceedings of IEEE conference on Computer Vision and Pattern Recognition (CVPR). Portland, OR, USA, 2013:2938-2945

[9] Alexis J, Olivier B. Random maximum margin hashing//Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Providence, RI, USA, 2011:873-880

[10] Gong Y, Lazebnik S. Iterative quantization: a procrustean approach to learning binary codes//Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Providence, RI, USA, 2011:817-824

[11] Josef S, Andrew Z. Video Google: a text retrieval approach to object matching in video//Proceedings of IEEE International Conference on Computer Vision (ICCV). Nice, France,2003:1470-1477

[12] Herve J, Matthijs D, Cordelia S. Hamming embedding and weak geometric consistency for large scale image search//Proceedings of European Conference on Computer Vision (ECCV). Marseille, France, 2008:304-317

[13] Herve J, Matthijs D, Cordelia S. Improving bag-of-feature for large scale image search. International Journal of Computer Vision, 2010, 87(3):316-336

[14] Mihir J, Herve J, Patrick G. Asymmetric Hamming Embedding: Taking the best of our bits for large scale image search//Proceedings of the 19th ACM international conference on Multimedia. Scottsdale, Arizona, USA, 2011:1441-1444

[15] Herve J, Matthijs D, Cordelia S. Packing bag-of-features//Proceedings of the IEEE 12th Conference on Computer Vision (ICCV). Kyoto, Japan, 2009:2357-2364

[16] Herve J, Matthijs D, Cordelia S. Product quantization for nearest neighbor search. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2011, 33(1):117-128

[17] Ge T, He K, Ke Q, et al. Optimized product quantization for approximate nearest neighbor search////Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Portland, OR, USA, 2013:2946-2953

[18] Chen Y, Guan T, Wang C. Approximate nearest neighbor search by residual vector quantization. Sensors. 2010, 10:11259-11273

[19] Jonathan B. Transform coding for fast approximate nearest neighbor search in high dimensions//Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR). San Francisco, CA, USA, 2010:1815-1822

[20] Babenko A, Lempitsky V. The inverted multi-Index//Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Providence, RI, USA, 2012:3069-3076

[21] Hwang Y, Han B., Ahn H. A Fast Nearest Neighbor Search Algorithm by Nonlinear Embedding//Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Providence, RI, USA, 2012:3053-3060

[22] David N, Henrik S. Scalable Recognition with a Vocabulary Tree//Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). New York, USA, 2006:2161-2168

AI Lie-Fu, born in 1985, Ph.D., E-mail: ailiefuhu@gmail.com. His research interests focus on content-based high-dimensional indexing and retrieval for large scale image.

YU Jun-Qing, born in 1975, Ph.D., professor, E-mail: yjqing@hust.edu.cn. His research interests focus on digital media processing and retrieval, multi-core programming environment.

GUAN Tao, born in 1978, Ph.D., associate professor, E-mail: qd_gt@126.com. His research interests focus on mobile visual search, augmented reality and computer vision.

HE Yun-Feng, born in 1977, Ph.D., lecturer, E-mail: yfhe@hust.edu.cn. His research interests focus on digital video processing and retrieval.

Background

The rapid development of internet and multimedia technologies leads to the explosion of multimedia information, especially picture and image, which, however, has placed people in an awkward situation where they can hardly get those favorite or similar images with accuracy and efficiency from a vast amount images unless those images are efficiently organized for browsing, searching and retrieval. Therefore, on the context of obtaining good search accuracy, how to retrieval similar images to query image rapidly, is important.

Given a query image, the retrieval methods generally first compute the distance from the query image to all the visual words in the corresponding inverted index, then, w nearest visual words is found and the visual features in the inverted lists, which associate to those visual words, are considered as the retrieval results. When only *knn* nearest retrieval results are needed, it needs to calculate the distances between the query feature and the retrieval results and a sorting procedure is followed. Actually, the features similar to query feature are lies in the position around the query feature. This is the issue that this paper will research on.

In this paper, two methods: exhaustive filtration and non-exhaustive filtration are proposed to filter query results for inverted indexing structure-based retrieval methods, where only the retrieval results around the query features are used for sorting. Exhaustive filtration constructs a hyper-sphere whose center is the query feature, and the corresponding radius is calculated adaptively. Only the features that lie in the hyper-sphere are used for sorting, then the number of features need to be sorted is reduced. Based on this, to reduce the time costs on filtering query results, non-exhaustive filtration partitions the inverted list into several sub-inverted lists, where the corresponding centroids are used to filter query results. To demonstrate the effectiveness of proposed methods, a typical method: residual vector quantization-based (RVQ) retrieval is used as an application example, which is combined with exhaustive filtration and non-exhaustive filtration respectively.

This research is financially supported by the National Natural Science Foundation of China (NSFC) under Grant No. 61173114, 61202300 and 61272202 and the Wuhan Application Foundation Research Project under Grant No. 2014010104010027.