文章编号: 1671-8860(2010)08-0920-05　　文献标志码: A

面向可变权值的多特征索引结构

何云峰[1]　于俊清[1]

(1　华中科技大学计算机科学与技术学院, 武汉市珞喻路 1037 号, 430074)

摘　要: 在基于样例的视频检索中, 视频数据采用多个高维特征数据描述, 针对不同的检索应用中这些特征数据的权值经常会发生变化的情况, 提出了一种面向可变权值的多特征索引树(multi-feature index tree)结构, 以满足用户在样例检索过程中对特征权值进行自定义的设置。多特征索引树采用适应于浏览的树型结构对视频的多个特征向量进行索引, 检索时, 通过遍历最低一层的集合节点, 以减少数据维数对检索效率的影响, 并针对多特征索引树结构, 提出了一种快速确定检索距离值的 ADD-*k*NN 检索算法。实验表明, 这种索引结构及相应的检索算法具有较好的性能。

关键词: 多特征索引树; 视频索引; 权值; *k*-NN 检索

中图法分类号: P208

基于内容的视频检索有语义检索和样例检索两种主要的方式[1]。由于视频数据的复杂性, 用于描述视频内容的低层特征有很多种, 并且这些特征大多数是高维数据, 为了提高检索的效率, 必须对这些高维数据建立索引。索引的目的是为了提高检索的效率, 但是当数据维度达到 30~50 维时, 索引结构实现的检索没有顺序检索的速度快, 即所谓的"维度灾难"问题[2,3]。为了解决这一问题, 国内外的专家主要从数据降维、数据近似和一维变换等三个方面进行了研究。典型的索引结构是 VA-File[4] (vector approximation file)。VA-File 采用近似向量来表述数据集中的每一个数据, 然后通过顺序遍历的方式访问近似向量文件, 获得检索结果。OVA-File[5] 在 VA-File 的基础上, 采用距离相近的近似向量就近放置的原则, 提高了数据检索的效率。一维变换是将高维数据映射为一维向量的算法, 包括 Pyramid-Technique[6]、iDistance[7] 和 iMinMax 等算法[8]。然而, 一维变换的过程必然带来信息的丢失, 可能导致在构造候选结果集时不能有效地排除不需要计算的数据对象, 降低了检索的效率。

随着多媒体数据库技术的发展, 以高维数据索引为基础的多特征索引结构也得到了广泛的关注。文献[8]采用单个 *M*-Tree 对所有的特征进行索引, 然后结合主成分分析与神经网络技术对数据进行降维。这一方面对大容量的数据库进行神经网络训练较为困难; 另一方面, *M*-Tree 结构对于超过 20 维的数据, 检索性能显著下降[9]。文献[10]提出利用降维方法和 B^+ 树索引结构实现多特征检索。本文提出了一种面向可变权值的多特征索引树结构。

1　多特征索引树(MFI-Tree)

1.1　相似度距离模型

与单个特征不同, 在权值可变的多特征检索应用中, 视频对象间的相似度距离是变化的。设 $F=(F_1, F_2, \cdots, F_n)$ 为用 n 个特征描述的视频内容, 其中 F_i 表示第 i 个特征, 它由 d_i 维向量组成。在多特征检索的方式下, 视频对象 P 和 O 之间的相似度距离为:

$$\begin{cases} \mathrm{Dist}(P, O) = \sum_{i=1}^{n} w_i \mathrm{Dist}_i(P, O) \\ \sum_{i=1}^{n} w_i = 1, w_i > 0 \end{cases} \tag{1}$$

式中, $\mathrm{Dist}_i(P, O)$ 表示第 i 个特征的相似度距离

收稿日期: 2010-06-15。

项目来源: 国家自然科学基金资助项目(60703049); 武汉市青年科技晨光计划资助项目(200850731353); 华中科技大学自主创新基金资助项目(M2009019)。

值; w_i 表示第 i 个特征在相似度计算时的权值。

同一个数据集中对象不同特征的距离处于不同的值域范围内, 因此, 需要进行归一化处理:

$$\mathrm{Dist}_i(P, O) = \frac{\mathrm{Dist}'_i(P, O)}{\mathrm{Dist}_{i\max}} \qquad (2)$$

式中, $\mathrm{Dist}'_i(P, O)$ 指根据第 i 个特征相应的相似度距离公式计算出的特征距离值; $\mathrm{Dist}_{i\max}$ 指在整个数据集中所有对象第 i 个特征的最大距离值, 这样, 两个对象所有特征的相似度距离值都归一化到[0, 1]之间。

在特征值归一化之后, 两个视频对象间的相似度距离值总是小于其 n 个特征相似度距离值中的最大值。因此, 在建立索引结构时, 两个对象的相似度距离用式(3)计算, 建立用于索引结构的相似度距离模型:

$$\mathrm{Dist}(P, O) = \max(\mathrm{Dist}_i(P, O)) \qquad (3)$$

从式(1)和式(3)可以看出, 当索引结构建立后, 相似度距离是最大特征距离值, 而检索时计算的距离值是加权后的相似度距离, 这意味着索引结构中的数据集合的覆盖半径较大, 而在检索时, 这些数据集合的覆盖半径会随着权值的变化而进行不同程度的收缩。

1.2　多特征索引树结构

与 M-Tree 等结构类似, 多特征索引树包括叶子节点和集合节点两类。叶子节点用于存储数据库中所有对象的数据, 其结构如下:

节点 ID	父节点 ID	特征 F_1	…	特征 F_n	特征距离值 d

其中, 父节点 ID 指当前叶子节点的父集合节点 ID; F_i 记录当前叶子节点第 i 个特征的特征值; d 是当前叶子节点与父集合节点中心对象的距离值。

集合节点用于存储多个叶子节点构成的数据库对象集合。与 M-Tree 的路由节点不同, 多特征索引树的集合节点并不用来检索, 而仅仅用于浏览应用, 以减少特征维度对树型结构检索效率的影响, 其结构如下:

节点 ID	父节点 ID	覆盖半径 R	集合中心对象 ID	浏览对象 ID	叶子节点数 LeafNum	结点标识 isRoot

其中, 节点 ID 和父节点 ID 用于构造树型的层次结构, 便于浏览; 覆盖半径指定集合中所有叶子节点与集合中心对象计算出的距离最大值; 集合中心对象 ID 指定记录集合节点中心对象数据的对象 ID; 叶子节点数 LeafNum 指定当前集合中包含的数据对象数目; 节点标识 isRoot 标明当前的集合节点是否是最低层的集合节点, 即不是其他任何集合节点的父节点的集合节点。由于构造集合节点的中心对象时, 为了使覆盖半径尽可能小会使用虚拟的节点对象, 因此, 集合节点中保存了用于代表集合节点的浏览对象 ID, 以提高浏览的响应速度。

1.3　索引结构的生成算法

对于多特征索引树而言, 其生成过程可以看成是一个数据集合不断分裂成多个小集合的过程。具体步骤如下:

1) 获得数据集合中单个特征的距离为最大值, 这是多特征索引树区别于其他动态索引结构的典型特征之一。

2) 数据集合分裂算法。考虑到浏览的应用, 这里的数据集合分裂算法是根据集合中数据的分布情况分裂为多个子集。如图 1 所示, 首先确定数据集合中距离值(式(3))最远的两个对象 A 和 B, 并将这两个对象分别插入到两个新子集中。然后分别计算数据集中所有对象与已有子集中第一个对象的距离值(式(3)), 确定其中距离最小的值 $d_{\min}$。如果 $d_{\min}$ 小于给定的阈值 AddDis, 表示当前对象与某个子集比较接近, 则将当前对象插入距离最小值对应的子集中, 如对象 K、L、H 和 G; 如果 $d_{\min}$ 大于给定阈值 NewSetDis, 表示当前对象与已有子集的距离都比较远, 如对象 C, 则新生成一个子集, 并将当前对象插入新子集中; 如果这两条都不满足, 则暂缓处理, 如对象 D、I 和 J。待所有的子集确定后, 根据距离最近原则插入相应的子集中。

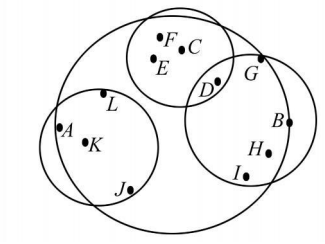

图 1　数据集合的分裂

Fig. 1　Set Divided in MFI-TREE

在所有的对象分配完毕后, 计算各个子集节点的数据, 取子集中距离最远的两个对象的中心点作为子集的集合中心, 然后计算子集中各个对象到子集集合中心的距离, 取距离最小的对象 ID 为数据集合浏览 ID, 取距离最大值(式(3))作为子集的覆盖半径。

在数据分裂算法中, 使用了两个阈值 AddDis 和 NewSetDis, 其中, AddDis 用于确定是否将对

象加入现有的子集中, 一般取 NewSetDis 的 1/2, 这里取0.35; NewSetDis 用于确定是否要新生成一个子集, 其取值为:

$$NewSetDis = \delta \cdot D_{max} \quad (4)$$

式中, D_{max} 表示进行分裂的集合节点的最大距离值; δ值用于确定集合分裂的程度, 为了避免子集数目过多, δ值一般在 0.6~0.8 之间, 这里取 0.7。

3) 对分裂后的子集进行判断, 如果满足分裂条件, 则继续分裂, 否则停止分裂。数据集合的分裂要考虑以下两个因素: 数据集合中的数据个数和数据集合的覆盖半径。在多特征检索中, 由于使用单个特征中最大的距离值作为两个对象的距离, 使得大部分的数据点都不能聚集在一个很小的半径值内(除了基本相同的对象外, 其他对象多个特征中总有一些会有较大的距离值), 因此, 主要考虑覆盖半径因素, 最大覆盖半径值宜在 0.1~0.3 之间, 这里取 0.3。

数据集合分裂算法由多个循环组成, 首先是生成两个初始子集, 其算法复杂度为 $O(3n)$; 然后遍历生成数据子集, 其算法复杂度在最坏情况下是 $O(n^2/2)$, 即此时待分裂的数据集合中所有的节点都足够分散; 最后生成新的节点, 其算法复杂度为 $O(n)$。因此, 一个数据集合分裂的算法复杂度为 $O(n^2)$。

1.4 插入与删除算法

视频对象的插入过程如下: 首先获得索引结构中最低层的所有集合节点, 然后将数据对象插入与中心对象距离(式(3))最近的集合节点, 并由下至上更新所有父集合节点中的叶子节点数。此时还需要判断数据对象与插入的集合节点中心对象的距离值是否大于覆盖半径, 如果是, 则修改当前集合节点的覆盖半径, 否则不需要修改; 最后对插入数据的集合节点进行分裂条件的判断, 若满足分裂条件, 则进行集合分裂操作, 否则插入完成。

数据删除算法如下: 首先找到需要删除的对象, 然后从下至上更新父集合节点中的叶子节点数; 对删除对象的集合节点进行更新操作, 如果集合节点的叶子节点数过少, 那么对该集合节点的父节点进行判断, 如果其仍然符合分裂要求, 则对其进行重新分裂; 如果不满足分裂要求, 直接删除其所有的子集合节点即可。

从索引结构的插入和删除算法过程来看, 最坏的情况就是会引起一次数据集合的分裂, 因此, 插入和删除算法的复杂度与分裂算法的复杂度相同, 都是 $O(n^2)$。

2 ADD-kNN 检索算法

建立索引结构的目的是为了提高检索的效率, 而检索效率的提高是由检索算法决定的。通常的视频信息检索主要有范围检索(range query)和 kNN 检索(kNN query)[11]两类。

与传统索引结构相比, 多特征索引树结构在采用基于过滤的方法时, 具有以下特点: ① 在多特征索引树中, 数据对象之间的相似度距离采用了特征中的最大距离值, 使得其集合节点大多具有较大的覆盖半径, 检索时, 采用过滤算法难以滤掉较多的数据集合; ② 在检索过程中, 叶子节点中保存的距离值一般比根据权值计算出来的相似度距离大, 使得检索时用于过滤的判断值过大, 不能有效地过滤掉不需要计算的数据, 降低了检索效率。为此, 本文提出了一种基于多特征索引树的 ADD-kNN (aggressive decided distance for kNN) 检索算法, 其主要思想是通过快速地减小用于过滤的判断值, 尽可能过滤掉大部分的数据, 以提高检索效率。其具体算法如下:

1) 根据用户选择的样例数据、特征及其权值, 构造检索对象。

2) 将检索节点列表和结果列表置空, 判断值 D 置为 0。

3) 直接遍历多特征索引树中最低层的集合节点, 降低维度对多特征索引树检索效率的影响: ① 计算检索对象与集合节点的权值距离(利用确定的权值) d, 通过集合对象的覆盖半径 R 计算检索对象与集合节点的最小距离 $D_{min} = d - R$; ② 如果 $D_{min} < 0$, 表示检索对象在集合内, 那么这个集合中的对象一定是要访问的对象(距离值最小为 0), 不再插入检索节点列表, 而是直接访问该集合的所有叶子节点; 根据已定义的权值计算叶子节点与特征对象的距离, 按照距离值由小到大的顺序插入到结果列表中。当然, 如果结果列表中已有 k 个数据, 而且当前插入对象的距离值大于判断值 D, 就不需插入; 处理结果列表, 将第 k 个数据的距离值赋给判断值 D, 并删除第 k 个以后的所有数据, 如果此时结果列表中不足 k 个数据, 则不更新判断值 D; ③ 如果 $D_{min} > 0$, 表示检索对象在集合外, 将集合节点的相关数据(包括最大最小值、节点 ID、集合中心对象 ID 等) 直接插入到检索节点列表中。

4) 遍历检索节点列表: ① 如果结果列表中的数据个数等于 k 个(多余的都被删除了), 且当前集

合的距离最小值大于判断值 D, 则不处理当前集合, 遍历检索节点列表中的下一个集合; ② 如果要处理当前集合, 则对当前集合的所有叶子节点进行遍历, 处理方法与步骤 3) 的第 ②步相同。

5) 输出结果列表。

从算法的复杂度来看, 检索算法在最坏情况下与顺序检索相同, 都是 $O(k+n)$, 其中, n 是数据集中的对象数, k 为需要检索出的对象数。而检索效率的提高主要依赖于算法中对数据集合的有效过滤。在 ADD-kNN 检索算法中, 利用多特征索引结构树的特点, 通过对一定需要遍历的数据进行相似度计算, 更早更快地降低用于过滤的判断值, 有效地过滤数据集合, 并且减少了排序过程的开销, 提高了检索的效率。

3 实验结果与分析

测试的数据集来自于 30 余部电影的 400 多个视频片段, 按照每个视频片段每 20 帧保存一帧的方法, 提取 1 万~ 3 万个视频帧图像, 按照 MPEG-7 标准中 12 维颜色布局描述符和 80 维边缘直方图描述符的提取方法和相似度距离计算公式[2], 提取视频帧图像的颜色和形状特征, 存储在关系数据库中。由于在检索过程中, 检索结果的准确性、正确性和完整性主要依赖于所选择特征的有效性, 因此, 本实验主要测试索引结构的有效性。实验环境采用 IBM T61 双核笔记本, CPU 主频 2 GHz, 内存 2 G, 操作系统为 Windows XP, 数据库采用 Oracle9i。

1) 索引结构的有效性测试。通过 6 幅从测试视频中提取的图片, 分别对 1 万、2 万和 3 万左右的视频关键帧分别建立 MFI-Tree 索引结构和 M-Tree 索引结构, 采用 ADD-kNN 算法进行 k-NN 检索(k= 20), 得到的测试结果如图 2 所示, 其中颜色布局特征和边缘直方图特征的权值分别为 0.6 和 0.4。可以看出, 使用 ADD-kNN 算法的 MFI-Tree 索引结构有较好的性能, 其平均相似度距离计算的次数明显少于 M-Tree 索引结构。而 M-Tree 由于在访问层次结构时会增加访问数据库的次数, 使得其检索性能下降。平均检索时间不仅与距离计算的次数有关, 还与索引结构的访问次数有关。

2) 可变权值检索的有效性。用不同的特征权值对 2 万幅视频关键帧对象进行检索。实验结果在颜色布局特征的权值为 0.4 时, 平均检索时间最长, 而平均相似度距离计算的次数在颜色布局特征权值为 0.3 时达到峰值。这是由于在建立索引结构中使用了两个特征距离值中的最大值, 因此, 在颜色布局或边缘直方图特征的权值接近 1 时, 两个对象加权的距离值接近索引结构中的距离值, 而数据集合中的对象大多集中在边缘上, 这样在检索过程中能够较快地降低过滤值 D, 提高检索效率。

3) k 值的影响。对 3 万幅关键帧图像进行不同 k 值的检索测试, 结果如图 3 所示。

从实验结果可以看出, 多特征索引树及其相应的 ADD-kNN 检索算法可以有效地实现多特征可变权值的检索, 且检索算法中用于过滤的距离值 D 的大小决定了数据库对象的访问量及其相似度距离的计算次数, 直接影响到检索的效率。而平均检索时间由于受系统运行状态的影响, 会产生误差, 有时这种误差足以抵消 ADD-kNN 检索算法的优势。

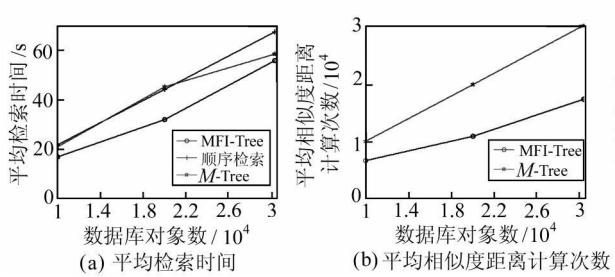

图 2　索引结构的有效性测试结果

Fig. 2　Result of Data Sizes

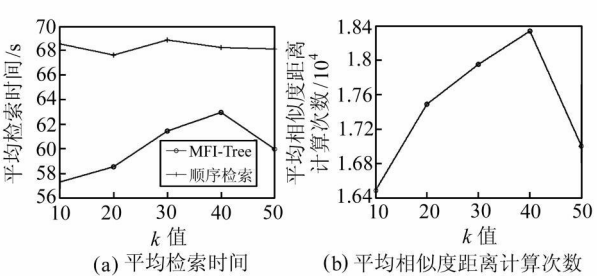

图 3　不同 k 值的实验结果

Fig. 3　Results of k Value

在进行视频特征提取时, 可以将关键帧图像另存为尺寸较小的缩略图, 存储在数据库中, 其 ID 与索引结构中的浏览对象 ID 关联。浏览时, 可以显示当前节点所有子节点的浏览对象 ID(集合节点) 或对象 ID(叶子节点) 对应的缩略图图像, 利用树型索引结构实现层次浏览。

由于特征相似度距离并不都是使用欧氏距离度量, 使得集合节点分裂后的子集仍然存在相互重

叠的问题,导致部分检索效率不高,因此,下一步的工作是在减少集合节点分裂后子集覆盖问题的基础上,着重解决以下两个问题:①在视频检索中,建立内容单元,包括场景、镜头等,建立在可变权值条件下的相似度距离模型,实现内容单元的检索;②大部分用于视频内容描述的特征不易被普通用户所理解,如何有效地建立面向可变权值视频检索的用户接口,更好地为用户服务,这是实现样例检索的关键。

参 考 文 献

[1] Lew M S, Sebe N, Djeraba C, et al. Content-based Multimedia Information Retrieval State of the Art and Challenges[J]. ACM Transactions on Multimedia Computing, Communications and Applications, 2006, 2(1): 1-19

[2] Hjaltason G, Samet H. Index-Driven Similarity Search in Metric Spaces[J]. ACM Transactions on Database System, 2003, 28(4): 517-580

[3] Bohm C, Berchtold S, Keim D A. Searching in High-Dimensional Spaces-Index Structures for Improving the Performance of Multimedia Databases[J]. ACM Computing Surveys, 2001, 33(3): 322-373

[4] Weber R, Schek H J, Blott S. A Quantitative Analysis and Performance Study for Simiilarity Search Methods in High-Dimensional Spaces[C]. The 24th VLDB Conference, New York, 1998

[5] Lu Hong, Ooi B C, Shen Hengtao, et al. Hierarchical Indexing Structure for Efficient Similarity Search in Video Retrieval[J]. IEEE Transactions on Knowledge and Data Engineering, 2006, 18(11): 1 544-1 559

[6] Berchtold S, Bohm C, Kriegel H P. The Pyramid-Technique: Towards Breaking the Curese of Dimensionality[C]. Int Conference on Management of Data ACM SIGMOD, Seattle, Washington D C, 1998

[7] Yu C, Ooi B C, Tan K L, et al. Indexing the Distance: An Efficient Method to KNN Processing[C]. The 27th VLDB Conference, Roma, Italy, 2001

[8] Ngu A H, Sheng Q, Huynh D, et al. Combining Multivisual Features for Efficient Indexing in a Large Image Database[J]. VLDB J, 2001, 9(4): 279-293

[9] Ciaccia P, Patella M, Zezula P. *M*-Tree: An Efficient Access Method for Similarity Search in Metric Spaces[C]. The 23rd VLDB Conference Athens, Greece, 1997

[10] Jagadish H V, Ooi B C, Shen Hengtao, et al. Toward Efficient Multifeature Query Processing[J]. IEEE Transactions on Knowledge and Data Engineering, 2006, 18(3): 350-362

[11] Chavez E. Searching in Metric Spaces[J]. ACM Computing Surveys, 2001, 33(3): 273-321

第一作者简介:何云峰,博士生,讲师,主要研究领域为基于内容的视频检索。

E-mail: yfhe@hust.edu.cn

Multi-feature Index Structure for Weighted Query Applications

HE Yunfeng[1] *YU Junqing*[1]

(1 School of Computer Science and Technology, Huazhong University of Science and Technology, 1037 Luoyu Road, Wuhan 430074, China)

Abstract: In the application of video retrieval by sample, video data is described by multiple high-dimensional features, and the weights of these features are changed in different queries. We propose a new indexing structure called multi-feature index tree(MFI-Tree) to index multiple high-dimensional features of video data for this retrieval application. MFI-Tree employs tree structure which is benefit for browsing application, and travels the last level aggregate node in retrieval application to improve the performance. And more, aggressive decided distance for *k*NN search algorithm which fast reduces the distance to prune the search space more effectively is proposed. The experimental results show that MFI-Tree and ADD-*k*NN algorithm offer performance advantages over sequential scan.

Key words: multi-feature index tree; video indexing; weighting; *k*-NN retrieval

About the first author: HE Yunfeng, Ph.D candidate, lecturer, majors in content-based video retrieval.

E-mail: yfhe@hust.edu.cn