동영상 내용기반 검색을 위한 고차원 벡터 데이터 색인 구조의 성능 분석+

이현조*, 장재우*, 박순영**
*전북대학교 컴퓨터공학과
**한국전자통신연구원 디지털홈연구단 인터넷서버그룹
e-mail:{hjlee, jwchang}@dblab.chonbuk.ac.kr
sunny@etri.re.kr

Performance Analysis of High-Dimensional Index Structure for Vector Data in Content-Based Video Retrieval

Hyun-jo Lee*, Jae-woo Chang*, Soon-Young Park**
*Dept of Computer Engineering, Chonbuk National University
**Internet Server Technology Group, Electronics and Telecommunications Reserach Institute

요 약

최근 멀티미디어 데이터, 특히 UCC를 중심으로 동영상 데이터가 급증하고 있다. 그러나 현재 대부분의 검색 시스템은 키워드 기반의 동영상 데이터 검색만을 지원하고 있으며, 따라서 사용자가 원하는 동영상 데이터를 효율적으로 검색하지 못하는 실정이다. 동영상 데이터에 대한 효율적인 검색을 지원하기 위해서는, 동영상의 내용(이미지, 색, 모양 등)을 고차원의 특징 벡터 데이터로 표현하여 유사한 동영상을 검색하는 내용-기반 검색이 요구된다. 본 논문에서는 내용-기반 검색을 위해 제안된 기존의 고차원 벡터 데이터 색인 구조를 실험을 통하여 성능을 비교하며, 이를 통해 동영상 내용-기반 검색에 가장 효율적인 색인 기법을 제시한다. 아울러 보다 효율적인 내용-기반 검색을 위한, 근사 k-NN 질의 탐색 기법의 유용성을 검증한다.

1. 서론

최근 멀티미디어 데이터, 특히 UCC를 중심으로 동영상 데이터에 대한 관심이 증가하고 있다. 해외 유명 UCC 동영상 사이트인 YouTube의 경우 하루 약 7만여 건의 동영상이 새로 등록되고 있으며, 하루 평균 600만명이 방문하고, 1억여 건에 이르는 동영상을 재생하고 있다. 이와 같이 동영상 데이터에 대해 많은 사람들의 관심이 쏠리면서, 그에 대한 효과적인 검색이 요구되어진다. 그러나 현재 동영상 데이터를 서비스하는 대부분의 인터넷 포탈 시스템은 단순한 키워드에 기반한 검색 서비스를 제공하고 있다. 이와 같은 단순 키워드-기반 검색은 정보 검색의 정확도(Precision)를 떨어뜨리며, 그 결과 사용자의 만족도를 저하시킨다. 동영상 데이터 검색의 정확도 및 사용자 만족도를 높이기 위해서는, 이미지, 색, 모양, 에니메이션, 또는 비디오 내의 주요 장면과 해당 장면에서의 객체의 움직임 등 동영상 데이터를 대표할 수 있는 내용을 기반으로 검색할 수 있어야 한다. 동영상 데이터의 특징들은 각각 벡터 데이터로 표현될 수 있으며, 따라서 동영상 데이터의 내용-기반 검색이란, 해당 동영상으로부터 추출한 고차원 특징 벡터 데이터(이하 고차원 데이터)를 이용하여 유사한 고차원 데이터를 지닌 동영상을 탐색하는 것이다.

본 논문에서는 동영상 데이터의 내용기반 검색을 위해 기존에 제안된 고차원 데이터 색인 기법들을 성능 비교하고, 실험 결과를 토대로 동영상 내용-기반 검색에 가장 효율적인 색인 기법을 제시한다. 아울러 보다 효율적인 검색을 위한 근사 k-NN 탐색 방안을 고찰한다. 본 논문의 구성은 다음과 같다. 먼저 2장에서는 기존의 고차원 벡터 데이터 색인 기법들에 대하여 소개한다. 3장에서는 고차원 벡터 데이터 색인 기법들의 성능을 분석하고, 근사 k-NN 탐색을 위한 방안을 제시한다. 마지막으로 4장에서는 결론 및 향후 연구에 대하여 기술한다.

2. 관련 연구

본 장에서는 기존 고차원 데이터 색인 기법들에 대하여 살펴본다. 기존에 제안된 고차원 데이터 색인 기법들은 크게 트리-기반 기법과 필터링-기반 기법으로 나누어진다. 트리-기반 기법으로는 X-Tree[1], SR-Tree[2], M-Tree[3], Pyramid-Tree[4] 등이 있으며, 필터링-기반 기법으로는 VA-file[5]과 CBF[6] 등이 있다. 트리-기반 기법 중 효율적이라고 알려진 M-Tree와 필터링-기

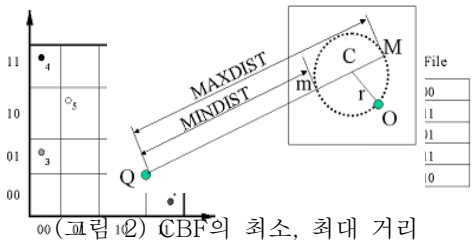
+ 본 연구는 정보통신부 및 정보통신연구진흥원의 IT신성장동력핵심기술개발 사업의 일환으로 수행하였음. [2007-S-016-01, 저비용 대규모 글로벌 인터넷 서비스 솔루션]

반 기법인 VA-file, CBF의 세 가지 기법에 대하여 살펴본다.

먼저 M-tree[3]는 고차원 데이터를 기반으로 효율적인 최근접점 탐색을 지원하기 위해 제안된 트리 구조이다. M-Tree의 노드 분할 방식은 첫째, 노드 V 내의 모든 데이터들 중, 서로 가장 멀리 떨어진 두 점을 찾고, 이 두 점을 잇는 선분 u에 노드 V 내의 모든 데이터를 투영하여, u의 이등분 점에 가장 가까운 점 A를 선택한다. 이때 점 A에서 u에 수직하는 선 L이 분할 기준선이 된다. 마지막으로, 기준선 L을 중심으로 LC(Left Child)와 RC(Right Child)로 노드 V을 분할한다. NN(Nearest Neighbor) 탐색은, 질의점 q가 주어지면 깊이우선탐색을 하여 후보 x를 찾고, 이때 질의 q와 후보 x의 거리를 r이라 한다. 다음, 질의 q로부터 거리 r 이내에, 방문하지 않은 나머지 자식 노드 C'의 맴버가 있는지 확인하고, 존재한다면 C'를 탐색한다. 트리-기반 고차원 데이터 색인 기법은, 차원이 증가함에 따라 성능이 급격히 저하하여 순차 탐색보다 성능이 떨어진다. M-tree의 경우 30차원 이상의 특징 벡터에 대해서 성능이 크게 저하된다. 한편 Hybrid Spill-tree[7]는 Google 의 동영상 내용기반 검색을 위한 고차원 벡터 색인 구조로 사용되고 있으며, M-tree 를 기반으로 구성되었다.

차원 증가에 따른 검색 효율의 감소를 해결하기 위해, R. Weber 는 VA-file[5] 을 제안하였다. VA-file은 특징 벡터와 시그니쳐(Signature)를 사용하여 필터링을 수행함으로써, 고차원 데이터에 대한 검색 성능을 개선한 방법이다. VA-file은 데이터의 구간을 여러 구역으로 나누고, 나누어진 각 영역에 bit를 할당하여 근사값을 생성한다. (그림 1)은 2차원 공간에서 VA-file이 생성된 모습을 보여준다. VA-file의 k-최근접점 탐색은 필터링 단계(Filtering step)와 벡터 접근 단계(Refinement step)로 나누어 수행된다. 필터링 단계에서는 VA-file을 순차 탐색하고, 질의점과 영역간의 거리를 이용하여 후보 벡터를 구한다. 즉, 질의 벡터가 주어지면 이를 질의 벡터 시그니쳐로 변환하고 VA-file을 순차 탐색하며, 이때 얻어진 k번째 최대 거리 값이 후보 영역 선택의 기준이 된다. 만일 시그니쳐 데이터가 기준 거리 내에 존재한다면, 이 근사 영역 내의 모든 벡터들은 후보 벡터가 된다. 벡터 접근 단계에서는 필터링 단계에서 선택된 후보 벡터의 실제 데이터에 접근하여 질의 벡터와의 거리와 현재 k번째 벡터와의 거리를 비교하여 결과 벡터를 구한다. VA-file은 필터링을 수행하여 일부 후보 벡터만을 탐색하므로 성능이 우수하다.

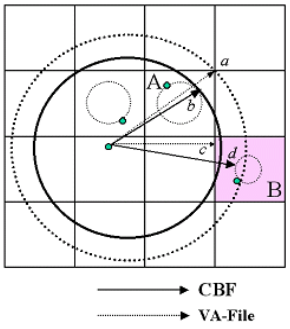
마지막으로 전북대학교에서 제안한 CBF(Cell Based Filtering) 기법[6]은 VA-file과 같은 필터링-기반 기법이나, 질의 지점에서 셀까지의 최소, 최대거리를 실체 객체와 더 근접하도록 설정하여 보다 우수한 필터링을 수행한다. 따라서 후보 벡터의 수가 감소되고, 검색 성능이 향상되었다. CBF에서 새로 정의한 질의 지점과 객체 사이



(그림 2) CBF의 최소, 최대 거리

(그림 1) Vector Approximation

의 최소 거리(MINDIST) 및 최대 거리(MAXDIST)는 객체와 객체가 저장된 셀의 중심 사이의 거리 r을 이용한다 (그림 2). 즉, 질의 지점 Q와 셀 중심 사이의 거리 O 에서 객체와 셀 중심 사이의 거리 r을 감산한 값을 최소 거리로, 질의 지점 Q와 셀 중심 사이의 거리 O 에서 객체와 셀 중심 사이의 거리 r을 합산한 값을 최대 거리로 정의하였다. (그림 3)는 CBF 의 필터링 효과를 나타낸다. VA-file의 경우에는 셀 B의 최소거리값 c가 k번째 최대거리값 b보다 짧으므로 B부분이 후보 셀로 선택되지만, CBF의 경우에는 셀 B의 최소 거리값 d가 k번째 최대거리값 b보다 더 크기 때문에 후보셀로 선택되지 않는다. 이로 인해 CBF 기법은 VA-File보다 효율적인 필터링을 수행한다.



(그림 3) 필터링 영역

3. 고차원 벡터 데이터 색인 구조의 성능 평가

3.1 성능평가 동기 및 환경

동영상 데이터에 대한 검색 정확도와 사용자 만족도를 높이기 위해서는, 단순한 키워드-기반 검색이 아닌 동영상의 특징에 기반한 내용-기반 검색이 요구된다. 동영상의 다수의 특징들은 고차원 데이터로 표현이 가능하며, 동영상 내용-기반 검색이란 고차원 데이터를 이용하여 질의와 유사한 데이터를 검색하는 것이다. 따라서 기존에 제안된 고차원 데이터 색인 기법의 성능을 비교하여, 가장 효율적인 색인 기법을 제시한다. 고차원 데이터 색인 기법에는 트리-기반 기법과 필터링-기반 기법이 있으며, 고차원 데이터를 다루는데 효율적인 방법으로 알려진 트리-기반의 M-Tree[3] 및 필터링-기반의 VA-file[5]와 CBF 기법[6]을 성능 비교한다. 성능평가는 삽입시간,

k-NN 탐색 시간, 저장공간 오버헤드(overhead)의 세 가지 항목을 통하여 수행한다. 아울러 보다 우수한 검색 성능을 달성하기 위해, 근사 k-NN 탐색을 위한 방안을 고찰하고, 그 유용성을 검증한다. 고차원 데이터 색인 기법의 성능 평가의 실험 환경은 <표 1>과 같으며, 실험에 사용된 데이터는 Corel_UCI[8] 리얼 데이터로 65차원의 66619개의 벡터 데이터(전체 크기: 57.8MB)로 구성되어 있다.

<표 1> 실험 환경

항목	성능
CPU	Intel Xeon 3.0Ghz
Memory	2GB
OS	Windows Server 2003
Compiler	VC++ 6.0

3.2 삽입시간 성능평가

<표 2>는 데이터 삽입 시간 성능을 나타낸다. VA-file과 CBF 는 몇 bit를 사용하여 시그니처를 생성하는지에 따라(4bit와 8bit) 성능을 측정하였으며 ()안의 숫자가 bit를 나타낸다. 실험 결과, 첫째, VA-file과 CBF의 필터링 기법이, M-tree에 비해서 30~83배 가량 성능이 우수함을 알 수 있다. 둘째, 필터링 기법인 VA-file과 CBF의 성능 비교에서는 CBF가 VA-file에 비해, 약 3배 가량 성능이 좋지 않음을 알 수 있다. 이는 CBF의 경우, 객체가 속해있는 셀의 중심에서 객체 사이의 거리를 구하는 추가 연산이 필요하기 때문이다. 마지막으로, VA-file과 CBF에서 시그니처 크기로 4bit와 8bit를 각각 사용하였을 때, 성능이 유사함을 알 수 있다.

<표 2> 삽입시간

색인 구조 명	삽입시간 (sec)
M-tree	316
VA-file(4)	3.8
VA-file(8)	4.2
CBF(4)	10.3
CBF(8)	10.4

3.3 k-NN 탐색 시간 성능평가

사용자 질의에 가장 가까운 k개의 데이터를 반환하는 k-NN 질의는 동영상 내용-기반 검색에 있어 가장 중요한 질의이다. 선택된 세가지 기법 모두, 정확 매치를 지원하므로 검색 결과의 정확도는 100%이며, k개의 검색 속도가 주요 평가 요소이다. 실험을 위해 k는 20과 50을 각각 사용하였고, 질의 데이터는 실제 사용된 데이터 내에서 임의로 추출하였으며, 하나의 질의 당 100번의 경과 시간을 측정하여 평균을 구하였다. <표 3>은 k-NN 탐색 시간의 결과를 나타낸다. 실험 결과, 첫째, CBF와 VA-file의 필터링 기법이 M-tree에 비해서, 2~10배 가량 성능이 우수함을 알 수 있다. 둘째, 필터링 기법인 CBF와 VA-file의 성능 비교에서는, CBF 기법이 VA-file보다 약 2배 정도의 성능이 우수함을 알 수 있다. 이는 CBF기법의 최소, 최대 거리 정책이 보다 효율적인 필터링을 보장하기 때문이다. 마지막으로, VA-file와 CBF 모두 8bit 시그니처가 4bit 시그니처보다 우수한 성능을 보였으며, 이는 8bit의 시그니처가 데이터를 좀 더 세밀하게 나누므로, 필터링 단계에서 더 많은 수의 데이터가 가지치기(pruning) 되기 때문이다.

<표 3> k-NN 질의 탐색 시간(WALL TIME)

색인 구조 명	탐색시간 (sec)	
	k=20	k=50
M-tree	0.788	0.914
VA-file(4)	0.408	0.426
VA-file(8)	0.168	0.178
CBF(4)	0.368	0.396
CBF(8)	0.079	0.081

3.4 저장공간 오버헤드 성능평가

<표 4>는 저장공간의 오버헤드를 나타낸다. M-tree의 저장공간 오버헤드가 154%로 가장 크다. 그 이유는 M-tree가 트리 구조이기 때문에, 고차원 데이터를 색인하기 위한 비단말 노드가 데이터의 양에 따라 증가하기 때문이다. VA-file과 CBF 기법은 시그니처 크기로 4 bit 사용시 약 32%, 8 bit 사용시 약 36% 이다. 시그니처의 크기가 4bit와 8bit 일 때 저장공간 오버헤드의 차이가 크지 않은 이유는, 시그니처 저장 파일의 크기가 데이터 파일에 비해 매우 작기 때문이다. 또한 CBF는 각 셀에서 객체까지의 거리가 VA-file에 비해 추가적으로 저장되지만, 그 크기가 상대적으로 작기 때문에 저장공간의 오버헤드는 VA-file과 비슷한 수준이다.

<표 4> 저장공간 오버헤드

색인 구조 명	저장공간(MB)	오버헤드(%)
M-tree	89.4	154.7%
VA-file(4)	18.9	32.7%
VA-file(8)	21.03	36.4%
CBF(4)	19.06	32.96%
CBF(8)	21.09	36.49%

3.5 근사 k-NN 탐색

본 절에서는 앞에서 성능 평가한 세 기법 중 필터링-기반 기법인 VA-file과 CBF에 대해 근사 k-NN(nearest neighbor) 탐색 방안을 고찰한다. VA-file과 CBF기법을 이용한 k-NN 탐색은 크게 두 단계로 수행된다. 첫째, 필터링 단계는 시그니처 값을 기반으로 후보셋을 생성하며, CBF의 경우 셀의 중심에서 데이터까지의 거리를 사용한 새로운 최소, 최대 거리를 통해 후보셋의 크기를 줄인다. 둘째, 벡터 접근 단계는 필터링 단계에서 얻어진 후보셋을 토대로 실제 벡터 데이터에 접근하여 질의와의 정확한 거리를 계산한다. 필터링 단계의 효과를 높이기 위해서, 시그니처 생성 크기는 8bit로 선정한다. 또한 벡터 접근 단계는 실제 벡터 데이터에 접근하는 I/O 횟수를 줄이기

위해 생략될 수 있다. 따라서 필터링에서 얻어진 후보 셋만을 정렬하여, 그 중 k*(1+ e) (e 는 근사 탐색을 위한 확장비율)을 반환하는 근사 k-NN 탐색 방법을 제시할 수 있다.

제시된 근사 k-NN 탐색 방안의 유용성을 검증하기 위하여, e의 값이 변할 때 검색된 결과 셋 가운데 올바른 결과 셋과 일치하는 비율인 정확도를 측정한다. <표 5>와 <표 6>은 k가 각각 20 과 50일 때의 근사 k-NN 탐색의 정확도를 나타낸다. 찾고자 하는 올바른 k개의 결과 셋과 근사 k-NN 탐색으로 얻어진 결과가 완전 일치할 경우 정확도는 100%이다. VA-file 과 CBF기법이 동일한 결과를 보이는 이유는 질의와 데이터 간의 거리가 시그니쳐를 사용한 근사값으로 계산되어, 두 기법에서 후보 셋을 각각 정렬할 때 결과 셋의 순서가 거의 동일하기 때문이다. 한편, k가 커지면, 근사 탐색을 위한 확장비율 e 가 작아도 100%의 정확도를 나타냄을 알 수 있다. 이를 통해, k값에 따른 적절한 확장비율을 선정함으로써, 근사 탐색으로도 100%에 근접하는 정확도를 얻을 수 있어 근사 k-NN 탐색의 유용성이 검증되었다.

<표 5> 정확도(k=20)

	e = 0	e = 0.25	e = 0.5
	k*(1+e) = 20	k*(1+e) = 25	k*(1+e) = 30
VA-file(8)	95%	99.5%	99.9%
CBF(8)	95%	99.5%	99.9%

<표 6> 정확도(k=50)

	e = 0	e = 0.20
	k*(1+e) = 50	k*(1+e) = 60
VA-file(8)	97.5%	100%
CBF(8)	97.5%	100%

3.6 성능 고찰

앞에서 삽입시간, 저장공간 오버헤드, k-NN 탐색 시간에 대해 M-tree, VA-file, CBF기법의 성능비교를 수행하였다. 성능비교 결과, M-Tree 가 가장 좋지 않은 성능을 나타냈으며, CBF 기법이 가장 우수한 성능을 나타내었다. M-Tree의 경우 CBF에 비해 삽입 시간은 30배, k-NN은 10배, 저장공간 오버헤드는 4배 정도 저하된 성능을 나타낸다. 이는 트리-기반 기법이 데이터의 차원이 커지면 검색 성능이 순차 탐색보다 나빠지는 'Dimensional curse'의 문제를 반영하고 있기 때문이다. 한편 가장 좋은 성능을 보인 CBF기법은, VA-file에 비해, 삽입 시간은 약 3배 정도 더 소요되나, 저장 공간 오버헤드는 거의 동일하였고, 검색 성능의 경우에는 약 1.5배 우수하였다. 대용량의 데이터 및 대규모 사용자를 고려하면 가장 중점이 되는 요소는 검색 성능이며, 따라서 고차원 데이터 색인 기법으로 가장 효율적인 기법은 CBF 기법이라고 할 수 있다.

아울러 보다 우수한 검색 성능을 달성하기 위해 근사 k-NN 탐색 방안을 고려하고 그 유효성을 검증하였다. 근사 탐색을 위한 확장비율 e 가 0.2~0.25 일 때, VA-file과 CBF 기법 모두 100% 에 근접하는 정확도를 달성하였다.

4. 결론 및 향후 연구

동영상의 개수가 기하급수적으로 증가함에 따라, 동영상의 특징을 고차원 벡터 데이터로 추출하여 이를 기반으로 검색하는 동영상 내용-기반 검색에 대한 관심이 고조되고 있다. 따라서 본 연구에서는 다음의 두가지 연구를 수행하였다. 첫째, 기존의 효율적인 고차원 벡터 데이터 색인 기법으로 알려진 M-tree, VA-file, CBF 기법의 성능 비교를 수행하여, CBF 가 가장 효율적인 인덱tm 기법임을 제시하였다. 둘째, 보다 효율적인 검색을 위한 근사 k-NN 탐색 방안을 제시하였으며, 제시한 방안이, 근사 탐색을 위한 확장비율 e를 0.2~0.25 로 설정하였을 때, 기존 k-NN 탐색 방법과 비교하여 약 99% 이상의 정확도를 유지함을 보였다. 향후 연구는 본 연구를 바탕으로 대용량의 고차원 데이터 검색을 효율적으로 지원할 수 있는 고차원 데이터 분산 저장 시스템을 설계 및 구현하는 것이다.

참고문헌

[1] S. Berchtold, D. A. Keim, H-P. Kriegel, "The X-tree : An Index Structure for High-Dimensional Data", Proceedings of the 22nd VLDB Conference, pp.28-39, 1996.

[2] Katayama N., Satoh S., "The SR-tree: An Index Structure for High-Dimensional Nearest Neighbor Queries", Proc. ACM SIGMOD Int. Conf. on Management of Data, pp. 369-380, 1997.

[3] P. Ciaccia, M. Patella, and P. Zezula. "M-tree: An efficient access method for similarity search in metric spaces." In Proc. of the Int. Conference on Very Large Databases,Athens, Greece, 1997.

[4] Berchtold S., Bohm C., Kriegel H.-P., "The Pyramid-Tree : Indexing Beyond the Curse of Dimensionality", Proc. ACM SIGMODE Int. Conf. on Management of Data, Seattle, 1998

[5] R. Weber, H.-J. Schek, and S. Blott, "A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces," In Proc. 24th Int. Conf. VLDB, pp. 194-205, 1998.

[6] 장재우, 한성근, 김현진, "셀 기반 필터링 방법을 이용한 고차원 색인 기법", 정보과학회 논문지 제 28권 2호 pp. 204~216, 2001년

[7] Ting Liu, Charles Rosenberg, Henry A. Rowley, "Clustering Billions of Images with Large Scale Nearest Neighbor Search", IEEE Workshop on Applications of Computer Vision, 2007

[8]http://kdd.ics.uci.edu/databases/CorelFeatures/CorelFeatures.data.html