KIPS Tr. Software and Data Eng.
Vol.3, No.6 pp.215~218 pISSN: 2287-5905

http://dx.doi.org/10.3745/KTSDE.2014.3.6.215

A Hashing Method Using PCA-based Clustering

Cheong Hee Park†

ABSTRACT

In hashing-based methods for approximate nearest neighbors(ANN) search, by mapping data points to k-bit binary codes, nearest neighbors are searched in a binary embedding space. In this paper, we present a hashing method using a PCA-based clustering method, Principal Direction Divisive Partitioning(PDDP). PDDP is a clustering method which repeatedly partitions the cluster with the largest variance into two clusters by using the first principal direction. The proposed hashing method utilizes the first principal direction as a projective direction for binary coding. Experimental results demonstrate that the proposed method is competitive compared with other hashing methods.

Keywords : Clustering, Approximate Nearest Neighbors Search, Principal Component Analysis, Hashing

PCA 기반 군집화를 이용한 해싱 기법

박 정 희†

요 약

해싱(hashing)을 기반으로 한 근사 최근접 이웃 탐색(approximate nearest neighbors search, ANN search) 방법에서는 데이터 샘플들을 k-비트 이진 코드로 변환하는 해쉬 함수들을 이용함으로써 근접 이웃 탐색이 이진변환 공간에서 이루어지게 된다. 본 논문에서는 PCA 기반 군집화 방법인 Principal Direction Divisive Partitioning(PDDP)를 이용한 해싱 방법을 제안한다. PDDP는 가장 큰 분산을 가지는 클러스터를 선택하여 그 클러스터의 첫 번째 주성분 방향을 이용하여 두 개의 클러스터로 분할하는 과정을 반복적으로 시행하는 군집화 방법이다. 제안하는 해싱 방법에서는 PDDP에서 분할을 위해 사용하는 주성분방향을 바이너리 코딩을 위한 사영벡터로서 사용한다. 실험결과는 제안하는 방법이 다른 해싱 방법들과 비교하여 경쟁력 있는 방법임을 입증한다.

키워드 : 군집화, 근사 최근접 이웃 탐색, 주성분 분석, 해싱

1. 서 론

큰 사이즈의 데이터 수집이 용이해짐에 따라, 데이터마이닝이나 패턴인식과 같은 다양한 분야에서 효과적으로 근접 이웃을 탐색할 수 있는 방법의 중요성이 커지고 있다. 그러나, 정확한 근접 이웃을 찾는 대부분의 알고리즘은 높은 메모리와 시간 복잡도를 가지고 있는 반면, 근사한 근접 이웃을 찾는 것만으로도 문제를 해결할 수 있는 경우도 많다[1, 2]. 최근에는 해싱(hashing)을 기반으로 한 근사 최근접 이웃(approximate nearest neighbors, ANN) 탐색 방법들이 제안되고 있다[3, 4, 5].

※ 이 논문은 2011년도 정부(미래창조과학부)의 재원으로 한국연구재단 기초연구사업의 지원을 받아 수행된 연구임(No. 2011-0007779).
† 정 회 원 : 충남대학교 컴퓨터공학과 부교수
Manuscript Received : March 21, 2014
First Revision : April 29, 2014
Accepted : April 29, 2014
* Corresponding Author : Cheong Hee Park(cheonghee@cnu.ac.kr)

해싱 기반 방법들에서는 데이터 샘플들을 k-비트 이진 코드로 변환하는 해쉬 함수(hash function)들을 이용함으로써 근접 이웃 탐색이 이진변환 공간에서 이루어진다. Locality sensitive hashing(LSH)[3]에서는 해쉬 함수로서 임의 사영(random projection)을 사용한다. 그러나 임의 사영을 사용하는 것은 높은 정확도(precision)를 얻기 위해 상대적으로 긴 이진 코드를 필요로 하게 되고 이는 낮은 재현률(recall)의 결과를 낳게 된다. LSH가 트레이닝 데이터에 독립적으로 해쉬 함수를 구하는 반면, 데이터를 이용하여 해쉬 함수를 얻는 방법에서는 데이터공간에서 가까이 있는 데이터 샘플들이 매핑 후에도 가까이 있도록 하는 매핑함수를 구한다. [4]에서 제안된 principal component analysis hashing(PCAH) 방법은 주성분분석(principal component analysis, PCA)을 적용하여 가장 큰 고유값을 갖는 주성분들을 선택하여 사영한 후 평균값을 이용한 임계치 처리(thresholding)에 의해 이진코드로 변환한다. Spectral hashing(SH)은 그래프 라플

라시안 고유벡터들의 라플라스-벨트라미 고유함수 (Laplace-Beltrami eigenfunction)로의 수렴에 바탕을 두고 있는 해싱 방법이다[5]. 그러나, k-비트 이진 코드로의 변환함수를 구하기 위한 실제 SH 알고리즘은 PCA를 이용하여 계산되어진다. SH에서 이상적으로는 다른 임계치 처리 방법을 가지고 같은 주성분이 여러 번 선택될 수 있으나, 어떤 특정한 상황에서는 각 주성분이 한 번씩 선택되게 되어 PCAH와 유사하게 된다.

이 논문에서는, PCA 기반 군집화 방법인 Principal Direction Divisive Partitioning(PDDP)[6]를 이용하여 근사 최근접 이웃 탐색에 활용 가능한 해싱 방법을 제안한다. PDDP는 PCA를 이용하여 클러스터들을 더 작은 사이즈의 클러스터들로 반복적으로 분할하는 분열 계층(divisive hierarchical) 군집화방법이다. 가장 큰 분산을 가지는 클러스터를 선택하여 그 클러스터의 첫 번째 주성분 방향을 이용하여 두 개의 클러스터로 분할하는 과정을 반복함으로써 전체적으로 균형이 맞는 분할을 추구한다. 제안하는 해싱 방법에서는 PDDP에서 분할을 위해 사용하는 주성분방향을 바이너리 코딩을 위한 사영벡터로서 사용한다. 2절에서는 관련연구를 살펴보고, 3절에서 PDDP를 이용한 해싱 방법을 제안한다. 4절에서 실험결과를 보이고 마지막으로 5절에서 결론을 맺는다.

2. 관련 연구

해싱에서는 가까이 있는 데이터 샘플들의 충돌(collision) 가능성을 멀리 떨어져 있는 것들에 대해서보다 크게 할 수 있는 해쉬 함수를 사용하는 것이 핵심이다. Locality sensitive hashing(LSH)에서는 데이터들이 p-stable 분포(예를 들어, 가우시안 분포)를 따른다고 가정한다[3]. 이 분포로부터 랜덤 사영벡터를 추출하여 해쉬함수 $h_i(x)=sign(w_i^T x+b_i)$ 를 구하고, k-비트 이진코드 $H=[h_1,\cdots,h_k]$ 를 구성한다. 그러나 높은 정확도를 얻기 위해 긴 이진 코드를 필요로 하게 되고 이는 낮은 재현률로 이어지게 되어, 이를 극복하기 위해서는 여러 개의 해쉬테이블을 사용하는 것이 필요하게 된다. 큐어리 q가 주어졌을 때, 각 해쉬테이블에서 q가 매핑되는 버킷의 원소들로 후보집합을 만든 후, 이 중에서 근접 이웃을 구하게 된다.

Principal component analysis hashing(PCAH)은 PCA를 이용하여 해쉬함수들을 구한다[4]. 주어진 데이터 집합에 대하여 PCA는 최대 분산을 가지는 사영벡터를 공분산행렬의 고유벡터로써 구한다. 공분산행렬의 가장 큰 고유값 k개에 해당하는 고유벡터 $w_1,\cdots,w_k$ 가 PCAH에서 사영벡터로 사용된다. k-비트 바이너리 코딩은 사영공간에서 평균값에 의한 임계치 처리로 얻어진다.

Spectral hashing(SH)에서는 $X=\{x_1,\cdots,x_n\}$ 로부터 k-비트 바이너리 코드 $\{H(x_1),\cdots,H(x_n)\}$ 로의 매핑이 다음 조건을 만족한다[5].

$$\begin{aligned} &\text{min}imize: \sum_{i,j} sim(x_i,x_j)\|H(x_i)-H(x_j)\|^2 \\ &subject\ to\ H(x_i)\in\{-1,1\}^k,\ \sum_i H(x_i)=0, \\ &\frac{1}{n}\sum_i H(x_i)H(x_i)^T=I \end{aligned} \tag{1}$$

식(1)의 NP-hard 문제는 제한조건 $H(x_i)\in\{-1,1\}^k$ 을 없애고 라플라스 고유함수에 의존하여 해결될 수 있다. 그러나 실제 SH 알고리즘은 PCA를 이용하여 다음의 세 단계로 구현된다.

1) PCA를 이용하여 데이터의 주성분들을 구한다.
2) 각 주성분 방향으로 사영된 데이터에서 k개의 작은 고유값에 해당하는 일차원 해석적 고유함수(analytic eigenfunction)를 구한다.
3) 전체에서 가장 작은 k개의 고유값에 해당하는 고유함수를 구한 후, 바이너리 코드를 얻기 위해 고유함수를 0에서 임계치 처리한다.

근사 최근접 이웃 탐색을 위해 해쉬테이블을 이용하는 대신 매핑된 이진코드공간에서 큐어리에 대해 가장 가까운 해밍 거리(hamming distance)를 가지는 K개의 원소를 리턴함으로써 해밍 거리를 이용한 근접 이웃 탐색을 수행하기도 한다.

3. PDDP를 이용한 해싱

PDDP는 전체 데이터를 한 개의 클러스터로 설정한 후 반복적인 클러스터의 분할을 통해 군집화를 하는 하향식 계층적 군집화 방법이다[6]. 한 클러스터를 두 개의 클러스터로 나누기 위해 가장 큰 분산을 갖는 주성분을 이용한다. 가장 큰 분산을 갖는 주성분, 즉 제1주성분은 그 클러스터에 있는 데이터로 구성되는 공분산행렬의 가장 큰 고유값에 해당하는 고유벡터 w를 의미한다. c를 클러스터 내의 데이터의 평균이라 할 때, 클러스터 내의 데이터 샘플 x는 $w^T(x-c)$ 의 부호에 따라 두 개의 클러스터로 나누어진다. 분기할 클러스터의 선택은 클러스터의 응집도에 대한 임의의 척도를 사용할 수 있다. PDDP에서는 이러한 척도들 중 하나로 평균을 0으로 이동시킨 데이터행렬의 Frobenius norm으로 정의되는 클러스터의 분산을 이용한다. 반복적으로 가장 큰 분산을 가진 클러스터를 분할함으로써 결국에는 비슷한 수의 원소들을 가진 클러스터들로 구성되는 군집화를 이루고자 한다.

우리는 해싱을 위한 사영 방향으로서 PDDP 수행과정에서 얻게 되는 주성분 방향을 사용한다. i번째 분기를 위해 선택된 클러스터의 제1주성분과 평균이 각각 w_i와 c_i라고 할 때, i번째 해쉬 함수로서 $h_i(x)=sign(w_i^T(x-c_i))$ 를 구성한다. k-비트 바이너리 코딩을 위해 전체 데이터 집합이 $k+1$개의 클러스터로 나누어질 때까지 분기를 계속함으로써 k개의 해쉬 함수를 얻을 수 있다. 제안된 알고리즘을 PDDPH(PDDP-based Hashing)라고 축약해서 표시하고, 그림 1에 요약하였다.

Input: $X=\{x_1,\cdots,x_n\}$: the set of data points k: the number of hash functions desired
Output: $\{h_i(x)=sign(w_i^T(x-c_i)):i=1,\cdots,k\}$ the set of hash functions
1. Initialize $T=\{X\}$ 2. for $i=1,\cdots,k$ 3. Select a cluster A with the largest scatter value in T 4. Compute a principal component w_i with the largest eigenvalue and mean vector c_i of the cluster A 5. Split A into two clusters according to $sign(w_i^T(x-c_i))$ and update T by removing A and adding two new clusters 6. end for

Fig. 1. PDDPH: A hashing method using PDDP

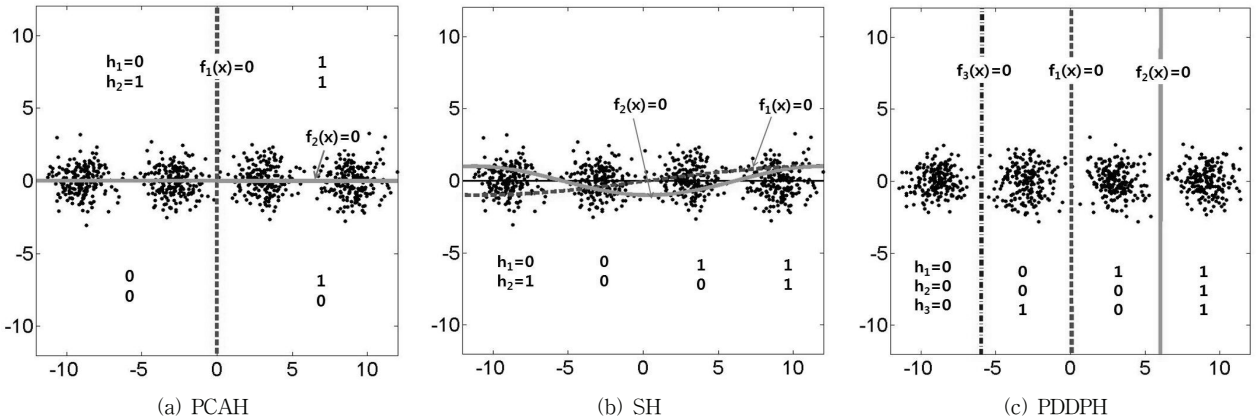
그림 2는 세 개의 방법 PCAH, SH, PDDPH를 적용한 예를 비교하고 있다. 표준정규분포를 따르는 2차원 데이터 샘플 200개를 생성하여 평균이 (-9,0)이 되도록 x축으로 평행이동하였다. 같은 과정으로 각각 평균이 (-3,0), (3,0), (9,0)이 되는 세 개의 집합을 더 생성한 후, PCAH, SH, PDDPH를 적용하여 바이너리 코드를 구성하였다. 그래프(a)에서 PCAH는 두 개의 주성분에서 얻어지는 경계함수 $f_i(x)=w_i^T(x-c_i)$에 의한 분할된 영역을 보여준다. 경계선에서 인접한 이웃이 다른 바이너리 코드를 가지는 경우를 보여준다. (b)에서 SH는 첫 번째 주성분방향으로 사영된 영역에서 두 개의 고유함수에 의해 네 개의 클러스터가 다른 코드로 매핑되는 것을 보여준다. (c)는 PDDPH로 얻어진 세 개의 해쉬 함수에 의한 3-비트 코드를 보여준다.

4. 실험 결과

제안된 방법을 LSH, PCAH, SH, DSH와 비교하였다. DSH(Density sensitive hashing)는 k-Menas 군집화에 기반한 해쉉 방법이다[7]. 성능 비교 실험을 위하여 두 개의 데이터 집합을 이용하였다. 첫 번째는 0부터 9까지 숫자를 손으로 쓴 70000개의 숫자이미지들로 구성된 MNIST 데이터이다[8]. 사이즈 28x28의 그레이 레벨 이미지로서 각 이미지는 784차원의 벡터로 저장된다. 데이터를 69000개의 트레이닝 데이터와 1000개의 테스트 데이터로 나눈 후, 트레이닝 데이터를 이용하여 해쉬함수들을 구하여 모든 데이터를 바이너리 코드로 변환한다. 테스트 집합에 있는 각 데이터를 큐어리로 사용하여 가장 작은 해밍 거리를 가진 500개의 데이터를 트레이닝 데이터로부터 구한다. 각 큐어리에 대해 정확도와 재현률을 구한 후, 모든 큐어리에 대해 평균을 취함으로써 성능을 평가한다. 큐어리에 대한 근접 이웃은 논문[7]에서처럼 숫자 라벨을 이용하여 같은 클래스의 원소로서 정의한다.

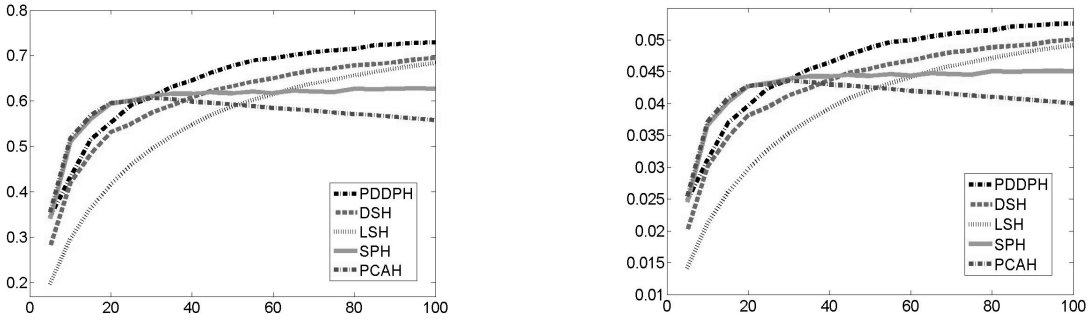
트레이닝 데이터와 테스트 데이터로 나눔을 임의로 30번 되풀이하여 실행하여 평균정확도(왼쪽 그래프)와 재현률(오른쪽 그래프)을 그림3에 나타내었다. x축에 있는 숫자는 k-비트 코드 길이인 k값을 나타내며 5부터 100까지 5씩 증가시키면서 성능을 평가하였다. 10에서 20 사이의 짧은 코드 길이에서는 SH와 PCAH가 좋은 성능을 나타내나, 30 이상의 코드 길이를 이용할 때는 제안한 방법이 다른 방법들에 비해 더 나은 성능을 보였다. 이는 그림2에서도 볼 수 있듯이 PDDPH에서는 충분한 수의 클러스터로 분할됨이 필요하기 때문이다.

실험에 사용한 다른 데이터는 20 뉴스그룹으로부터 취한 약 20000개의 기사를 포함하는 20 newsgroup 데이터이다. 전체 기사에서 50번보다 적게 나타나는 term을 제거한 후, 6165차원의 19944개의 다큐먼트 벡터들을 가지게 된다. 데이터는 19000개의 트레이닝 데이터와 1000개의 테스트 데이터로 나눈다. 각 큐어리에 대해 가장 작은 해밍 거리를 가지는 200개의 다큐먼트를 트레이닝 데이터에서 추출하는 점을 제외하고 나머지 실험 환경은 첫 번째 실험과 동일하게 하였다. 그림4에서 30번의 실험을 반복하여 얻은 평균정확도와 평균재현률을 보이고 있다. 첫 번째 실험과 달리 PCAH 방법이 가장 좋은 성능을 보이고 있다. 이는 고차원 데이터에서 PCA에 의해 얻어지는 큰 분산에 해당하는 주성분 방향들의 직교성에 의해 비트들의 중복성(redundancy)를 효과적으로 줄일 수 있었기 때문이라 분석된다.



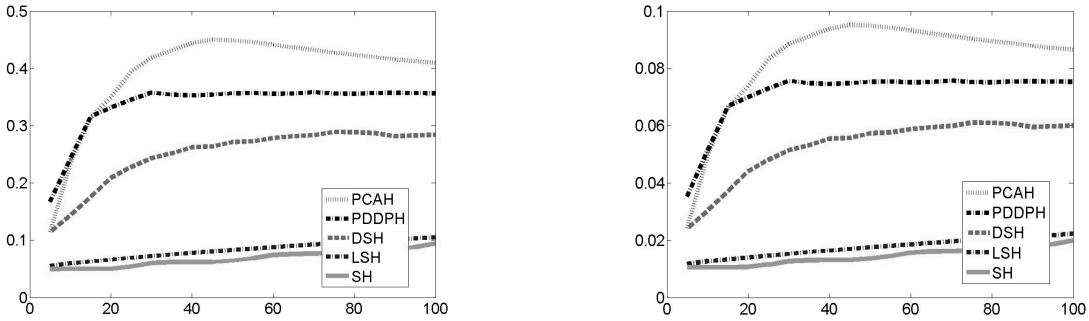
(a) PCAH (b) SH (c) PDDPH

Fig. 2. Illustration of binary encoding by PCAH, SH, and PDDPH



(a) Precision (b) Recall

Fig 3. Performance comparison using MNIST data



(a) Precision (b) Recall

Fig 4. Performance comparison using 20 newsgroup data

5. 결 론

PCA 기반 군집화 방법인 PDDP를 이용한 해쉉 방법을 제안하고, 다른 해쉉 방법들과 비교하였다. 실험결과는 제안한 방법이 약간 긴 코드 길이에서 다른 해쉉 방법들에 비해 나은 성능을 가짐을 보였고, 특히 k-Means를 이용한 군집화를 이용한 해쉉방법에 대해서도 경쟁력이 있음을 보였다.

Reference

[1] P. Indyk and R. Motwani, “Approximate nearest neighbors: Towards removing the curse of dimensionality”, *Proceedings of ACM Symposium on theory of computing*, 1998.

[2] M. Muja and D. Lowe, “Fast approximate nearest neighbors with automatic algorithm configuration”, *Proceedings of International Conference on Computer Vision Theory and Applications*, 2009.

[3] A. Gionis and P. Indyk and R. Motwani, “Similarity search in high dimensions via hashing”, *Proceedings of VLDB*, 518-529, 1999.

[4] X.-J. Wang and L. Zhang and F. Jing and W.-Y. Ma, “Annosearch: Image auto-annotation by search”, *Proceedings of CVPR*, 1483-1490, 2006.

[5] Y. Weiss and A. Torralba and R. Fergus, “Spectral Hashing”, *Proceedings of Advances in Neural Information Processing Systems*, 21, 1753-1760, 2008.

[6] D. Boley, “Principal direction divisive partitioning”, *Data mining and knowledge discovery*, 2(4), 325-344, 1998.

[7] Y. Lin and D. Cai and C. Li, “Density sensitive hashing”, *Proceedings of CoRR*, 2012.

[8] http://yann.lecun.com/exdb/mnist

박 정 희

e-mail : cheonghee@cnu.ac.kr

1998년 연세대학교 수학과(박사)

2004년 University of Minnesota, Computer Science & Engineering(박사)

2005년~현 재 충남대학교 컴퓨터공학과 부교수

관심분야 : Data Mining, 패턴인식