Hashing Techniques

윤성의 (Sung-Eui Yoon)
Associate Professor
KAIST

http://sglab.kaist.ac.kr

Class Objectives

- Understand the basic hashing techniques based on hyperplanes
- Get to know a recent one based on hyperspheres

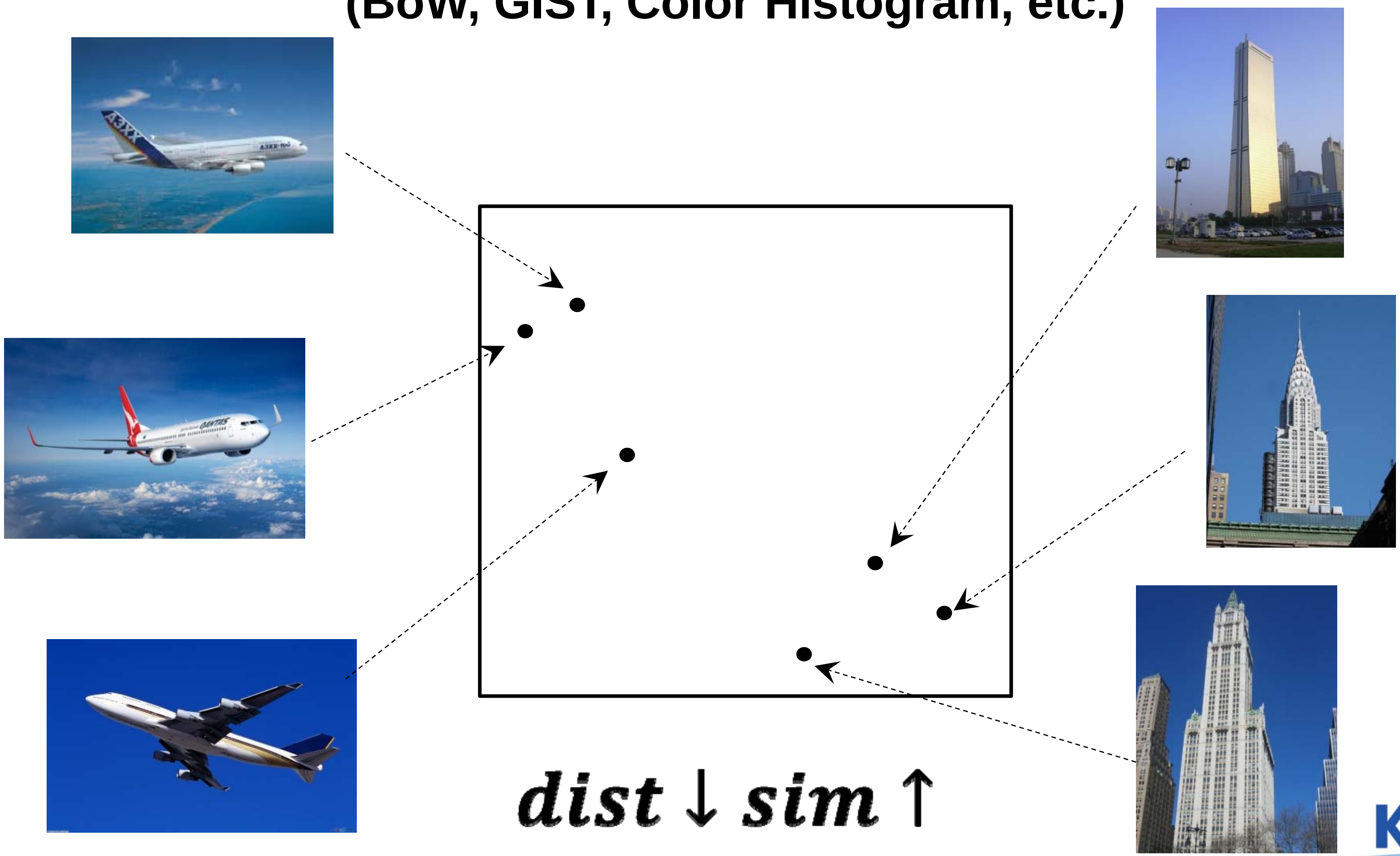
Image Retrieval

Finding visually similar images

Image Descriptor

High dimensional point

(BoW, GIST, Color Histogram, etc.)

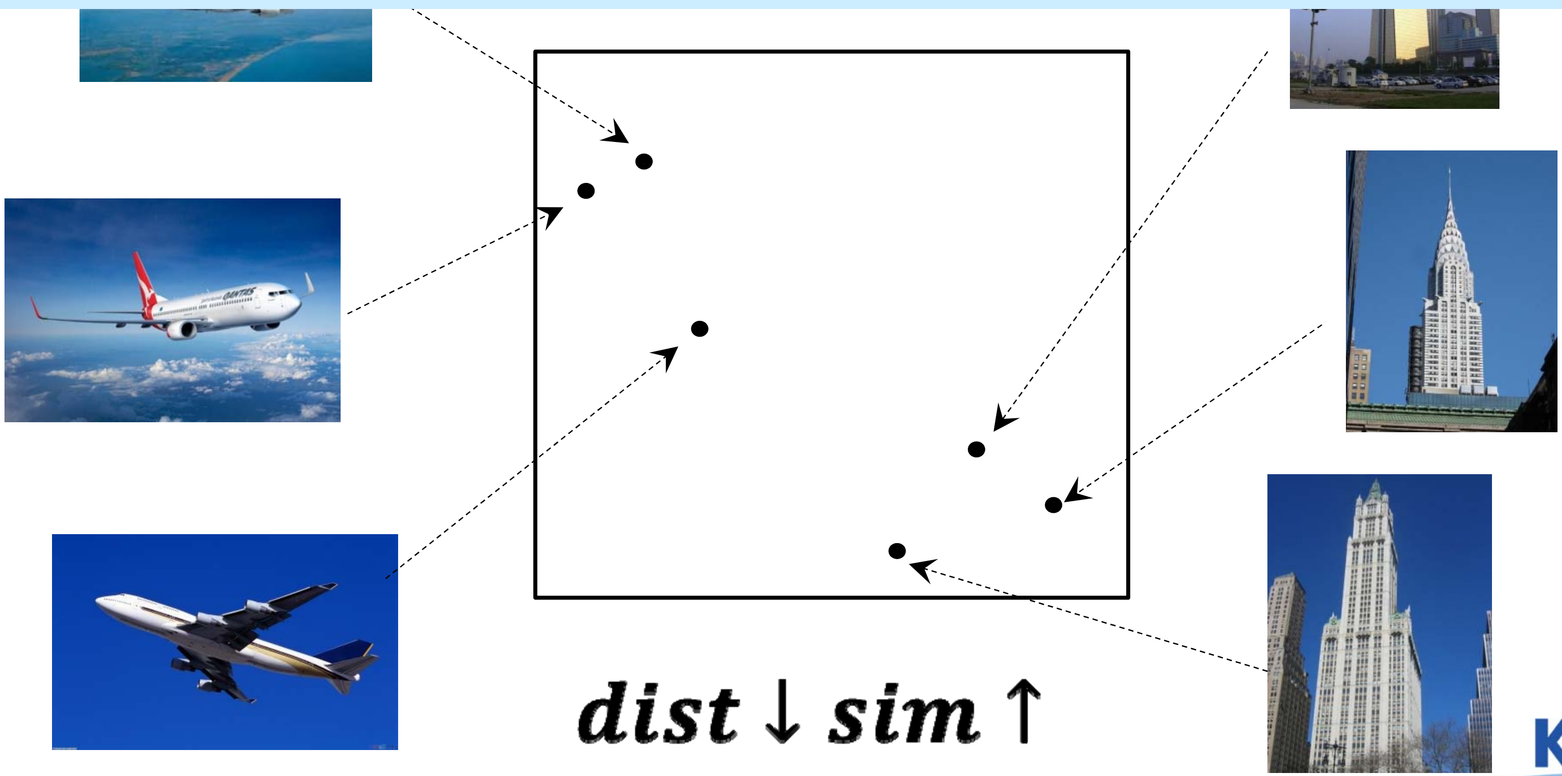


$$dist \downarrow sim \uparrow$$

Image Descriptor

High dimensional point

Nearest neighbor search (NNS) in high dimensional space



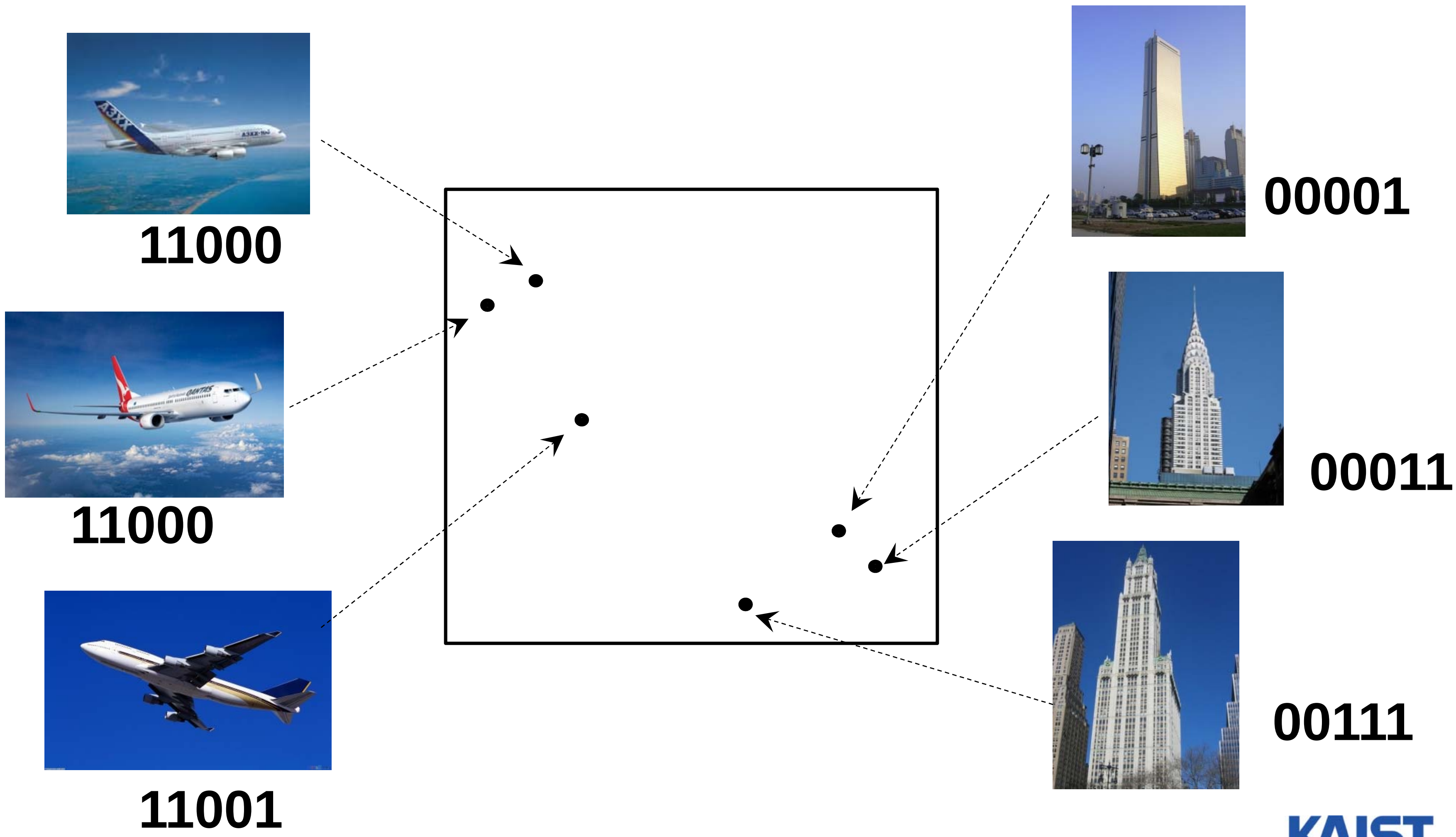
Challenge

	BoW	GIST
Dimensions	1000+	300+
1 image	4 KB+	1.2 KB+
1B images	3 TB+	1 TB+

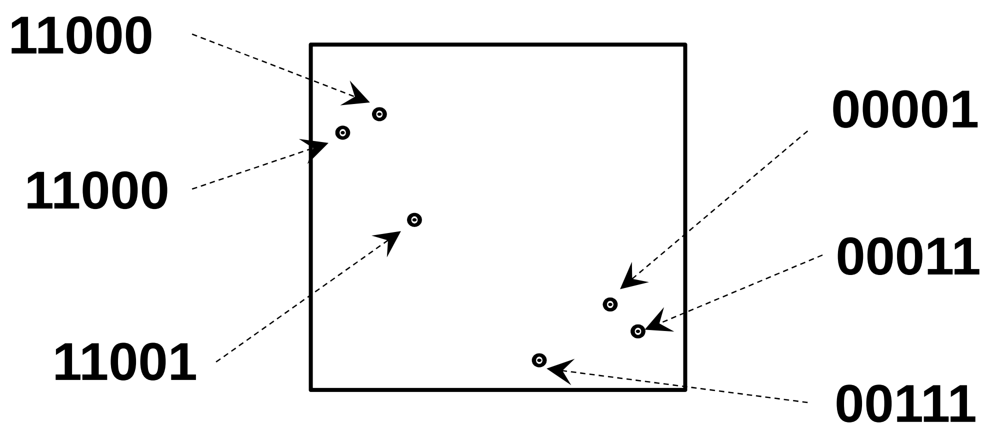
$$\frac{144\ GB\ memory}{1\ billion\ images} \approx \frac{128\ bits}{1\ image}$$

KAIST

Binary Code



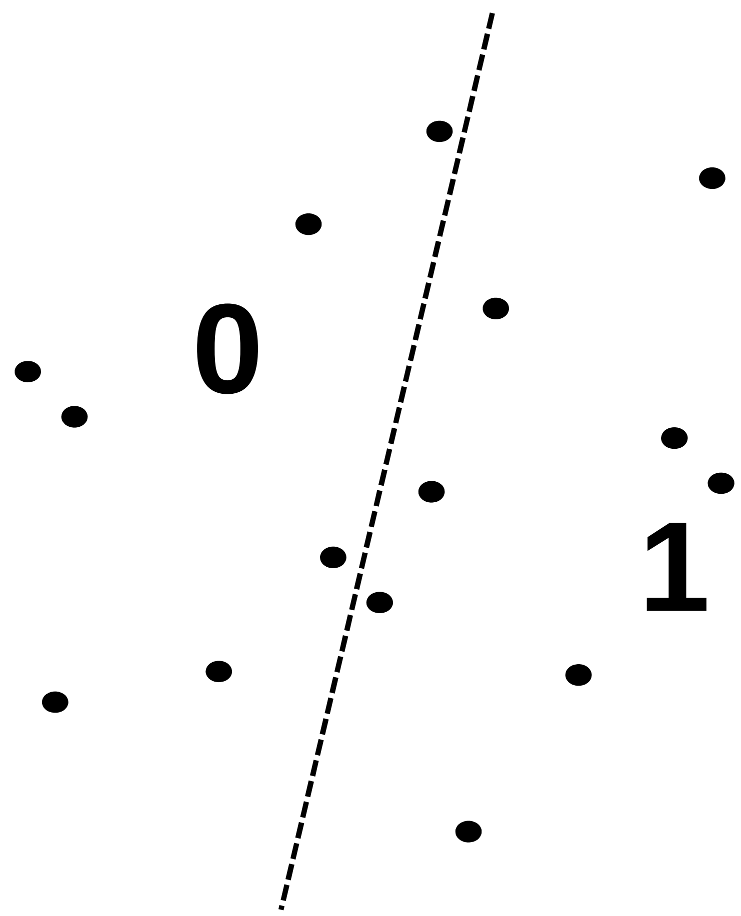
Binary Code



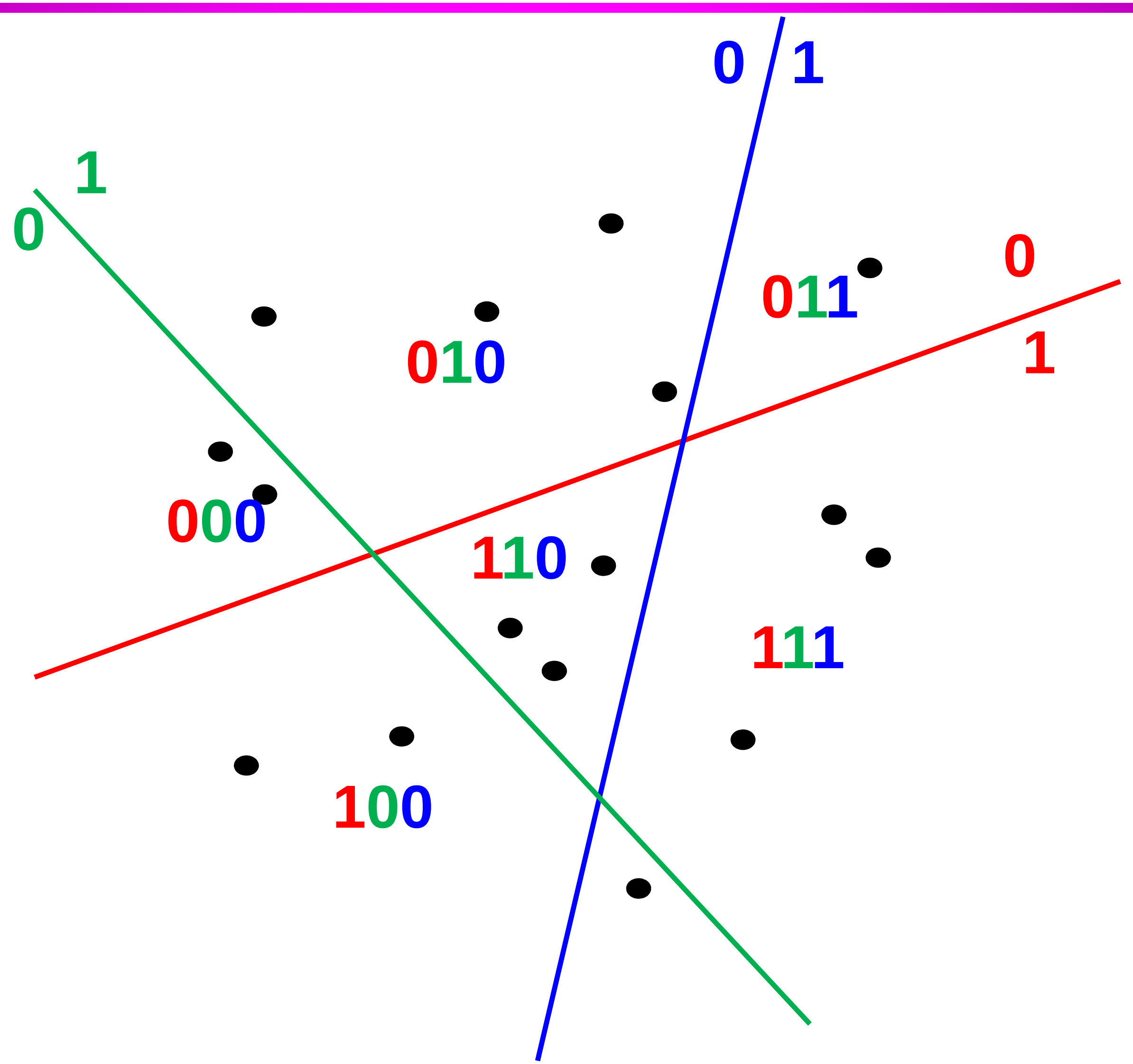
* Benefits

- Compression
- Very fast distance computation
  (Hamming Distance, XOR)

Hyper-Plane based Binary Coding



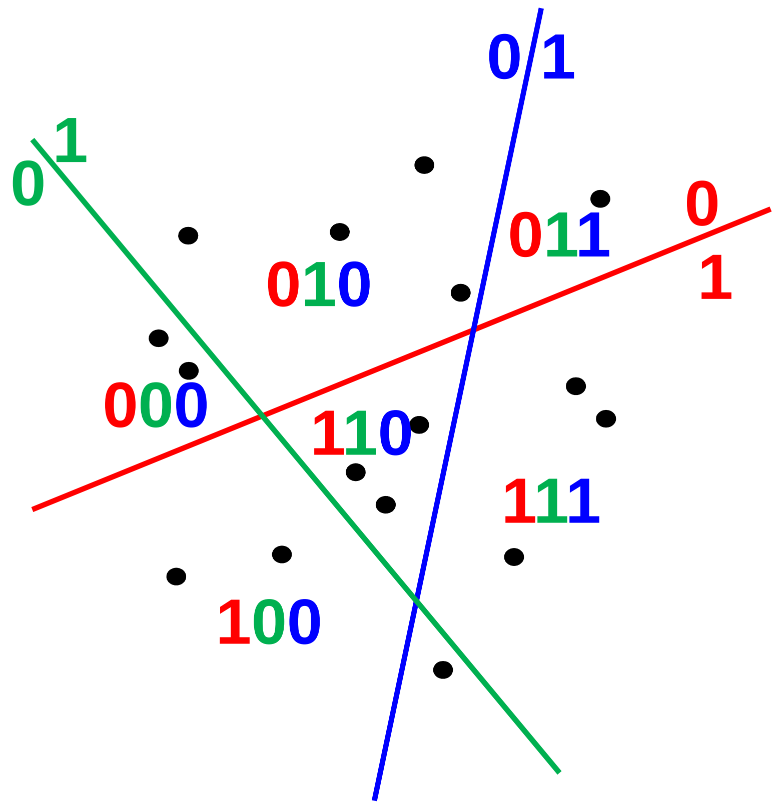
Hyper-Plane based Binary Coding



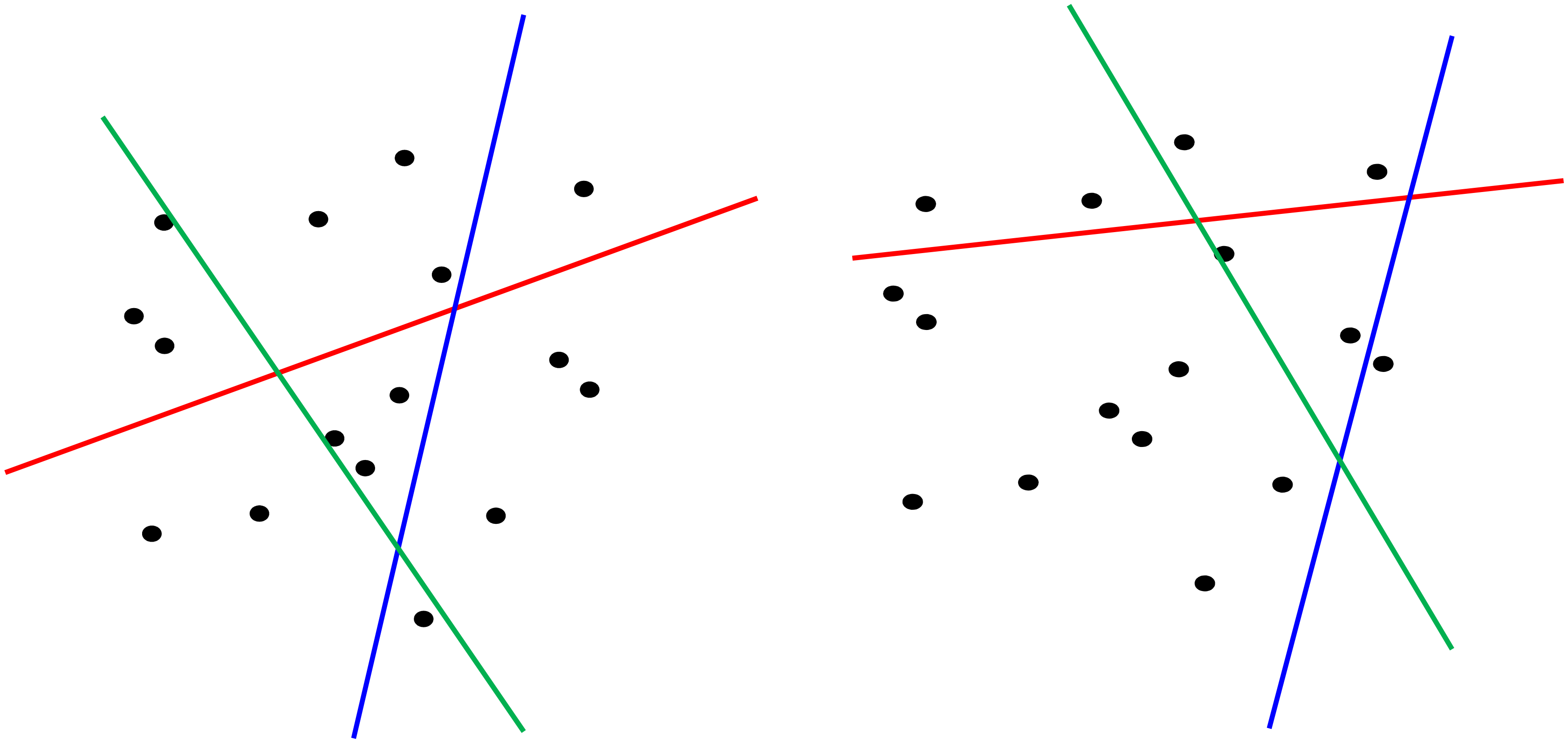
Distance between Two Points

- **Measured by bit differences, known as Hamming distance**
- **Efficiently computed by XOR bit operations**

$$d_{hd}(b_i, b_j) = |b_i \oplus b_j|$$



Good and Bad Hyper-Planes



Previous work focused on how to determine good hyper-planes

Previous Work

- **Random hyper-planes from a specific distribution**
  [Indyk STOC 1998, Raginsky NIPS 2009]
- **Spectral graph partitioning**
  [Yeiss, NIPS 2008]
- **Minimize quantization error**
  [Gong, CVPR 2011 oral session]
- **Independent component analysis**
  [He, CVPR 2011 oral session]
- **Support Vector Machine**
  [Joly, CVPR 2011]

Components of Spherical Hashing

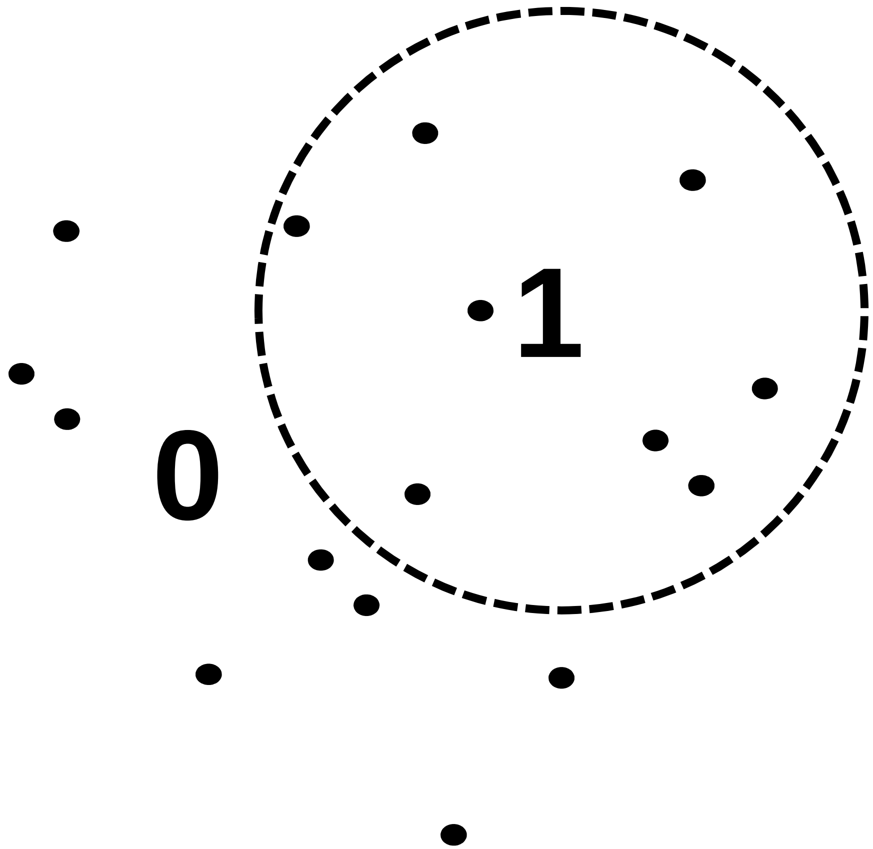
- **Spherical hashing**
- **Hyper-sphere setting strategy**
- **Spherical Hamming distance**

KAIST

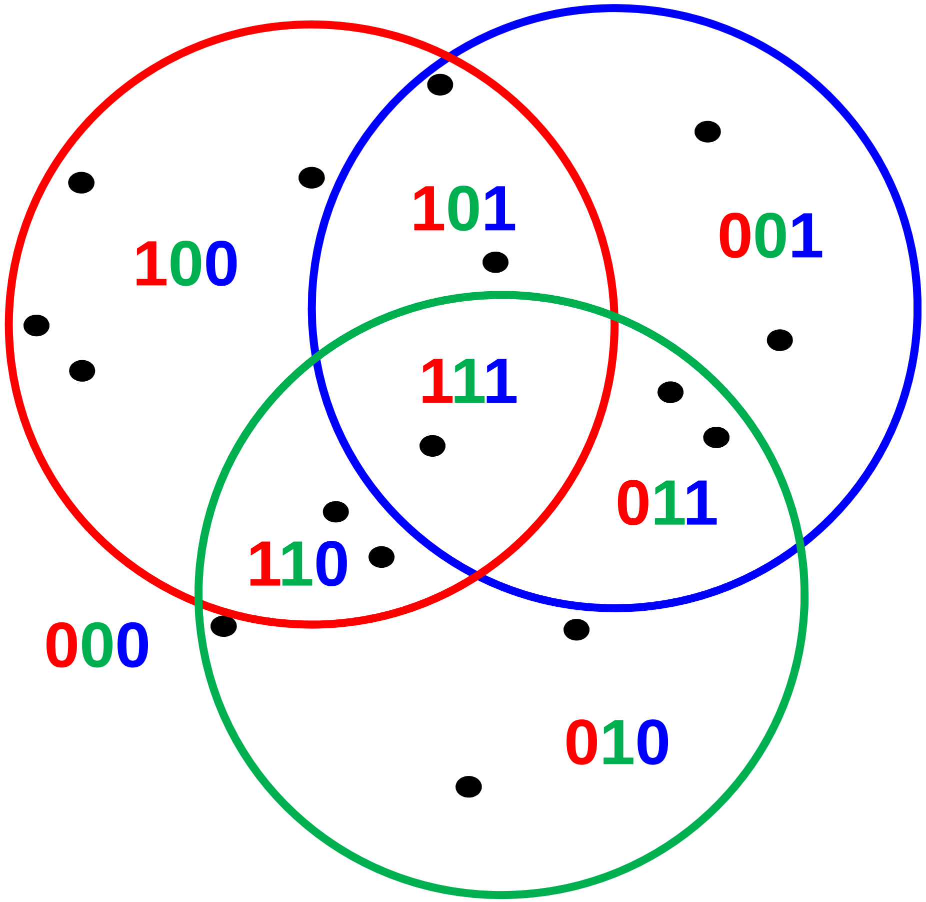
Components of Spherical Hashing

- **Spherical hashing**
- **Hyper-sphere setting strategy**
- **Spherical Hamming distance**

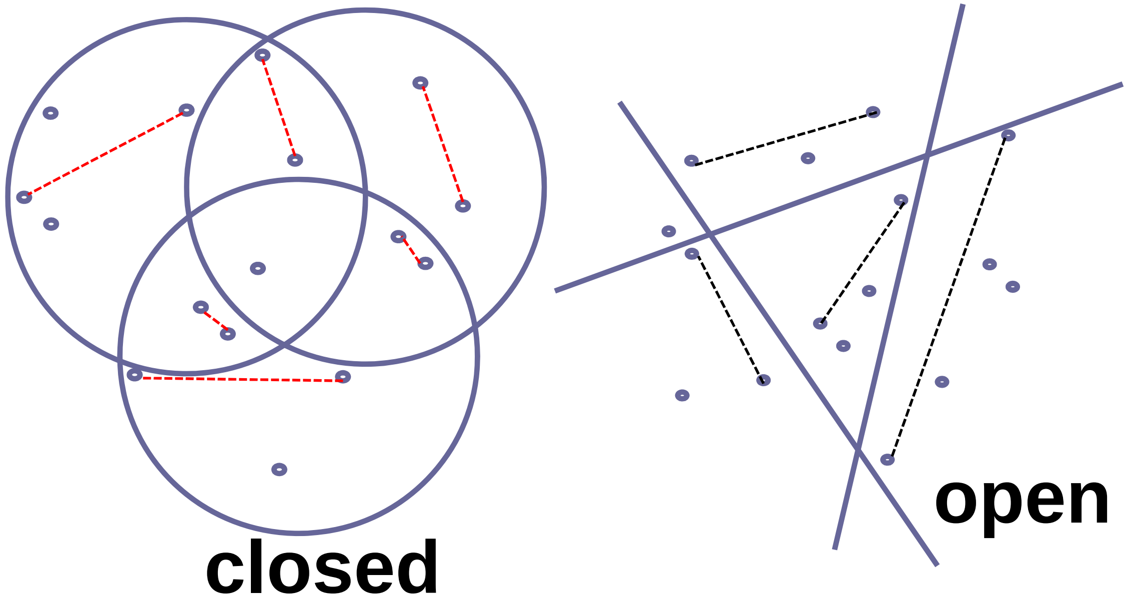
Spherical Hashing [Heo et al., CVPR 12]

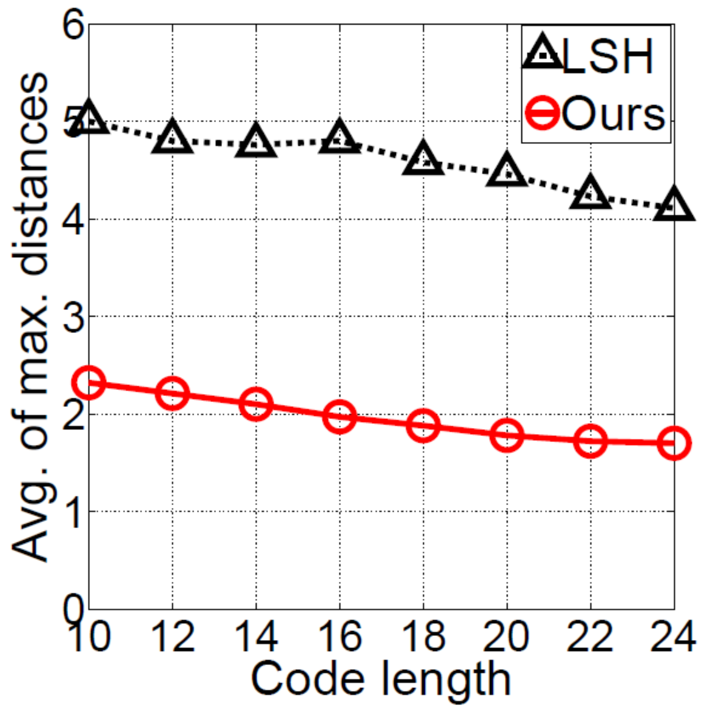


Spherical Hashing [Heo et al., CVPR 12]



Hyper-Sphere vs Hyper-Plane





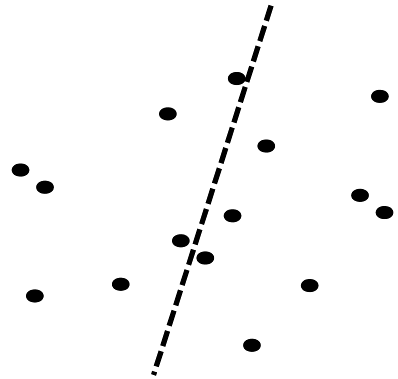
Average of maximum distances within a partition:
- Hyper-spheres gives tighter bound!

Components of Spherical Hashing

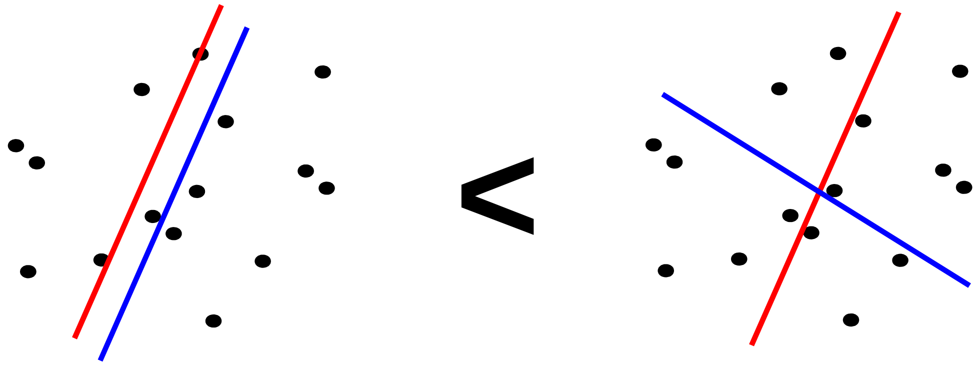
- Spherical hashing
- **Hyper-sphere setting strategy**
- Spherical Hamming distance

Good Binary Coding [Yeiss 2008, He 2011]

1. **Balanced partitioning**

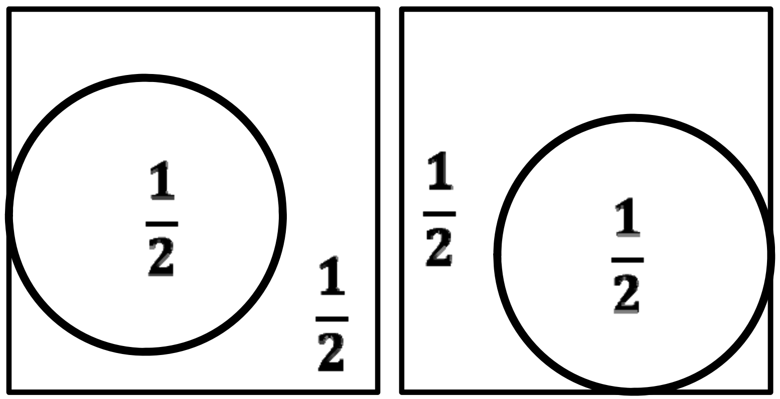


2. Independence

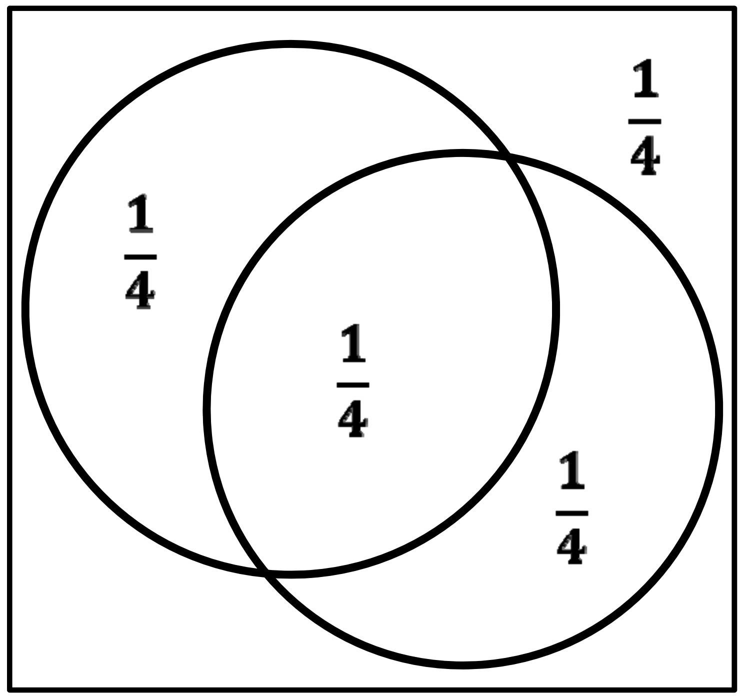


Intuition of Hyper-Sphere Setting

1. Balance



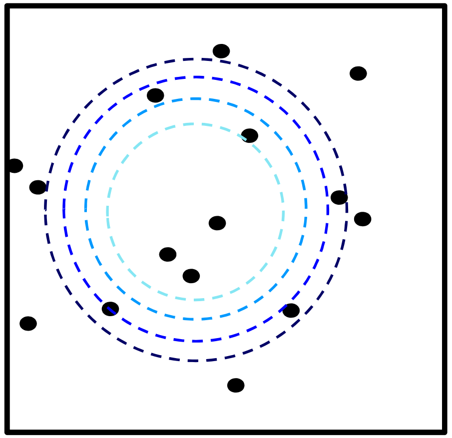
2. Independence



Hyper-Sphere Setting Process

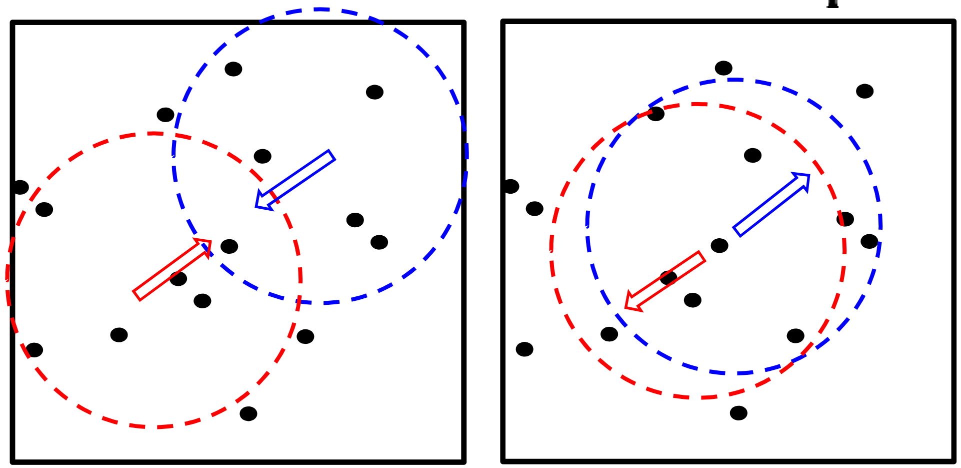
1. Balance

- by controlling radius for $n(S) = \frac{N}{2}$



2. Independence

- by moving two hyper-spheres for $n(S_1 \cap S_2) = \frac{N}{4}$

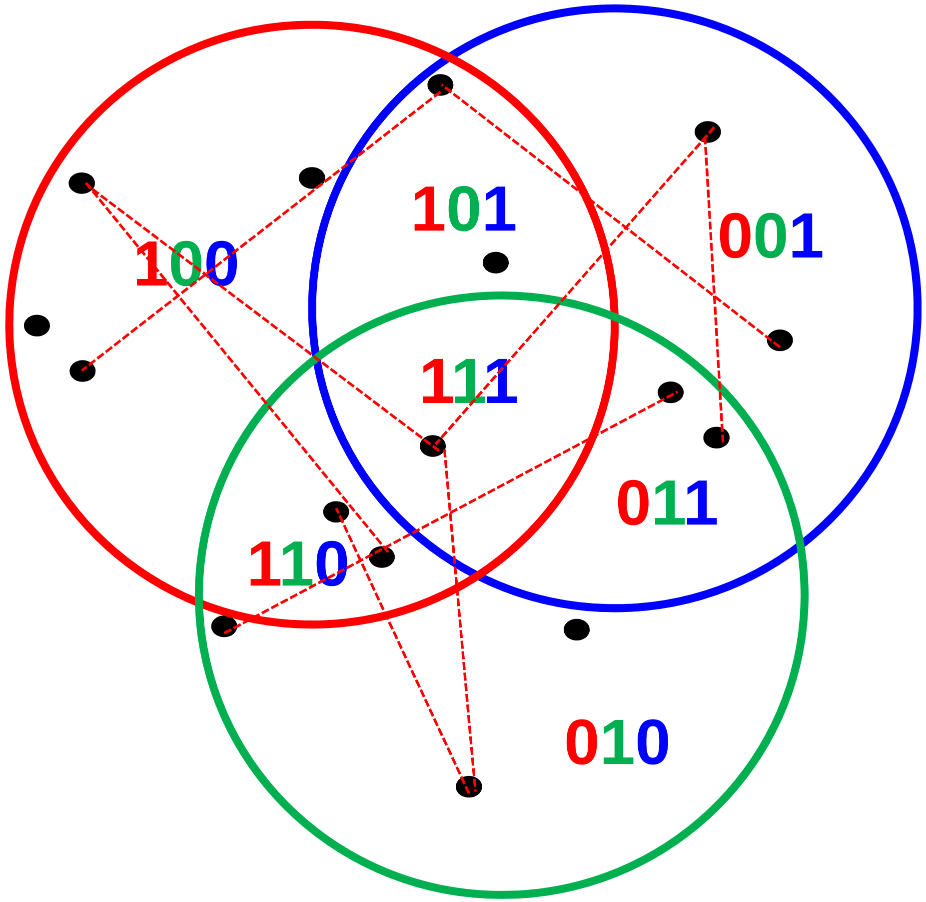


Iteratively repeat step 1, 2 until convergence.

Components of Spherical Hashing

- Spherical hashing
- Hyper-sphere setting strategy
- **Spherical Hamming distance**

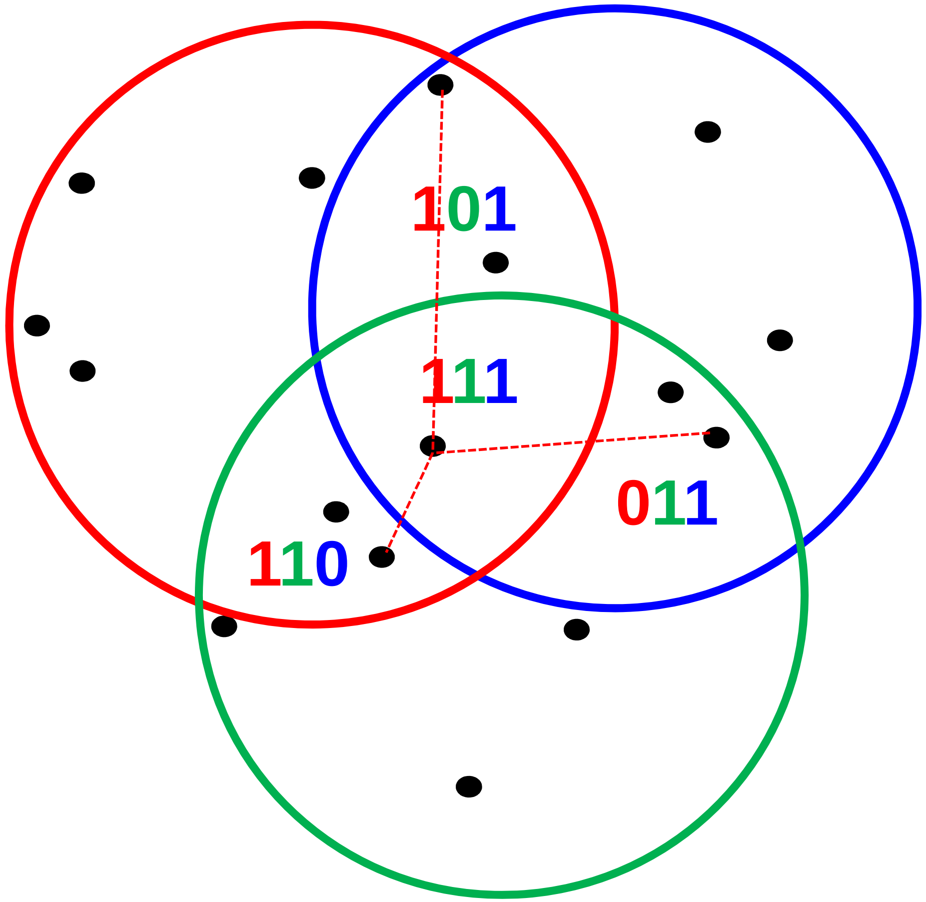
Max Distance and Common '1'



Common '1's

: 1

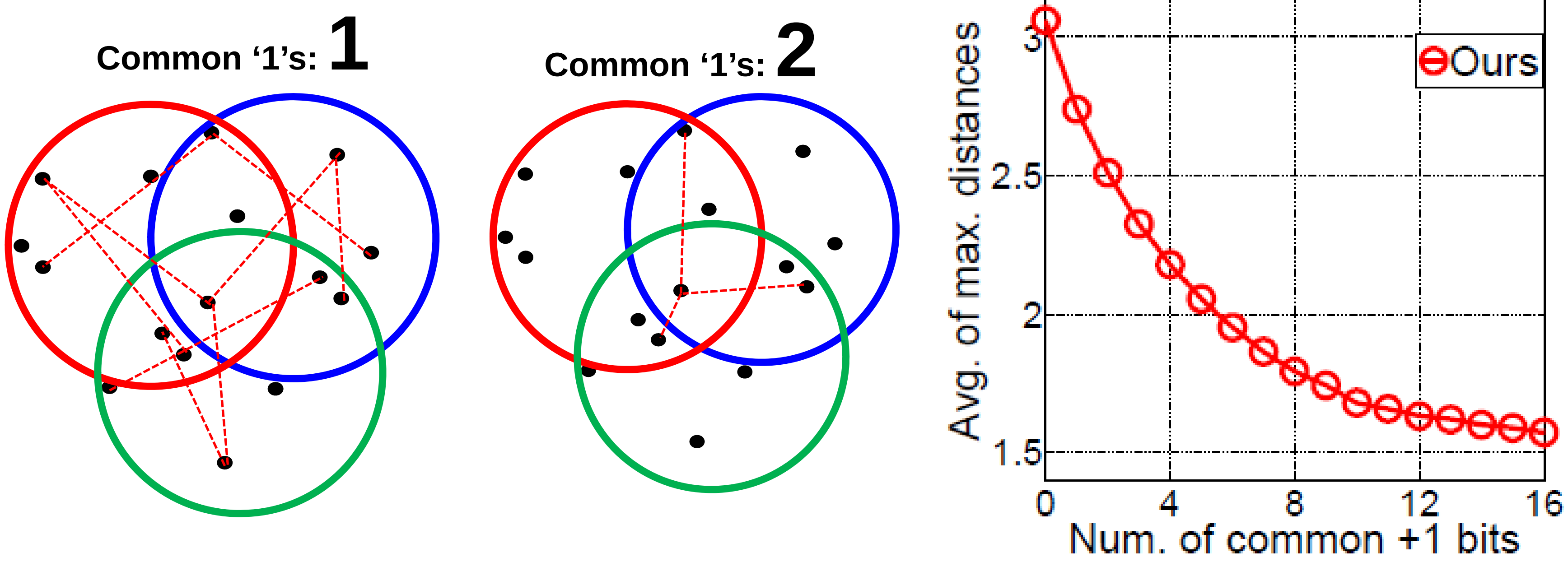
Max Distance and Common '1'



Common '1's

: **2**

Max Distance and Common ‘1’



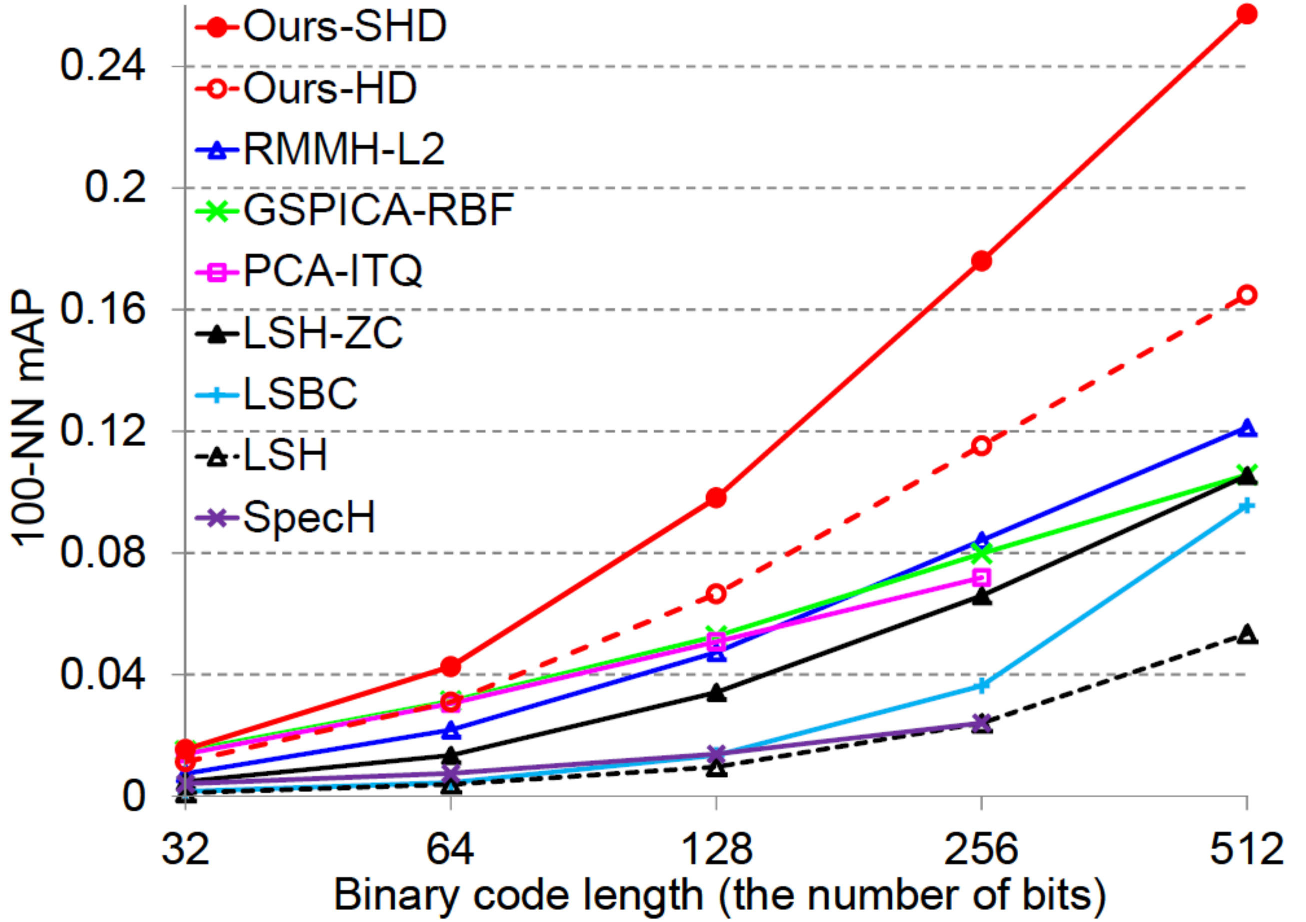
Average of maximum distances between two partitions: decreases as number of common ‘1’

Spherical Hamming Distance (SHD)

$$d_{shd}(b_i, b_j) = \frac{|b_i \oplus b_j|}{|b_i \wedge b_j|}$$

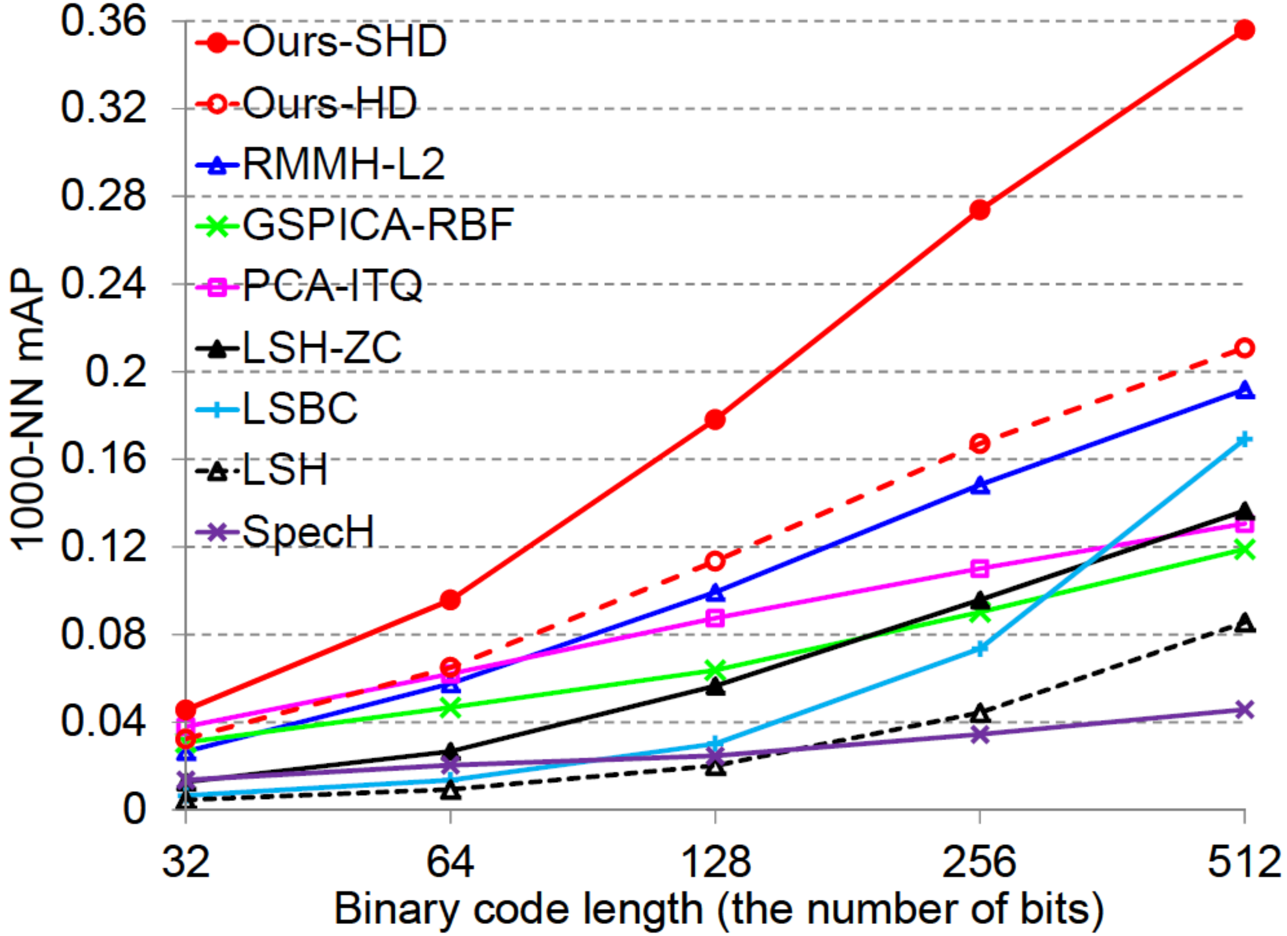
SHD: Hamming Distance divided by the number of common '1's.

Results



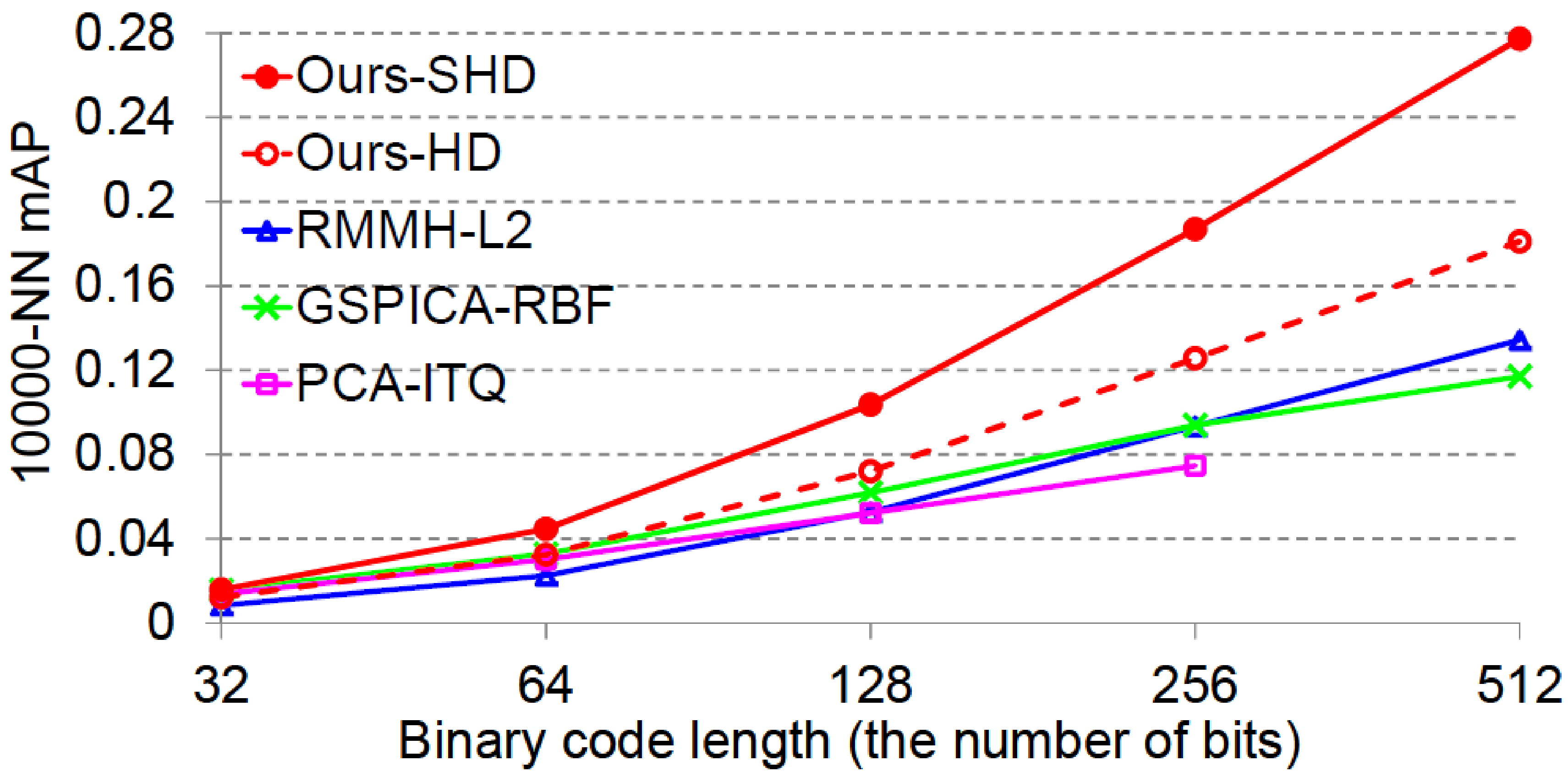
384 dimensional 1 million GIST descriptors

Results



960 dimensional 1 million GIST descriptors

Results



384 dimensional 75 million GIST descriptors

Summary

- **The need of binary code embedding**

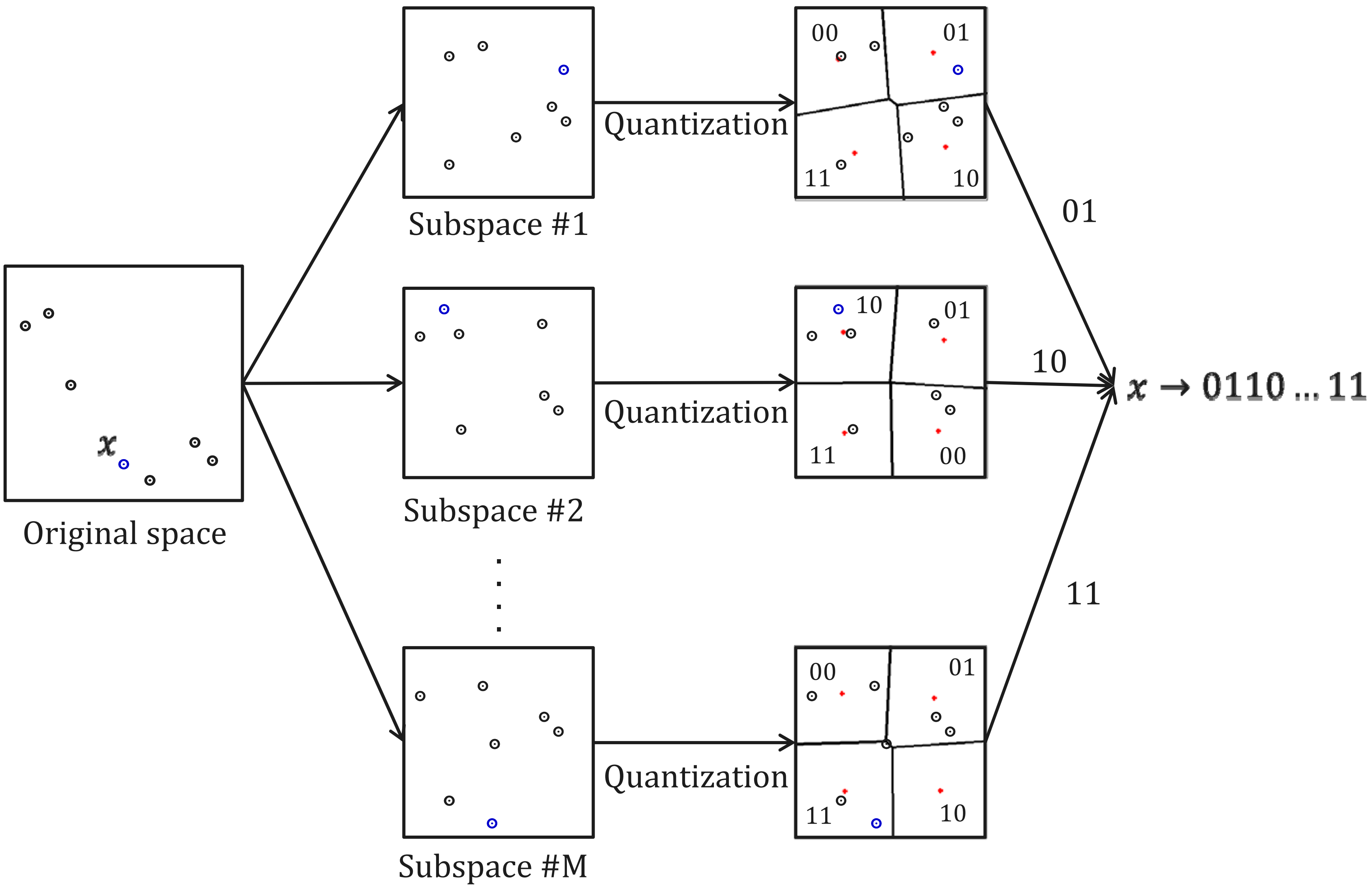
- **Spherical binary code embedding**
  - **Uses spherical hashing for tighter bounds**
  - **Iterative process to achieve balance and independence**
  - **Spherical Hamming distance**

Distance Encoded Product Quantization

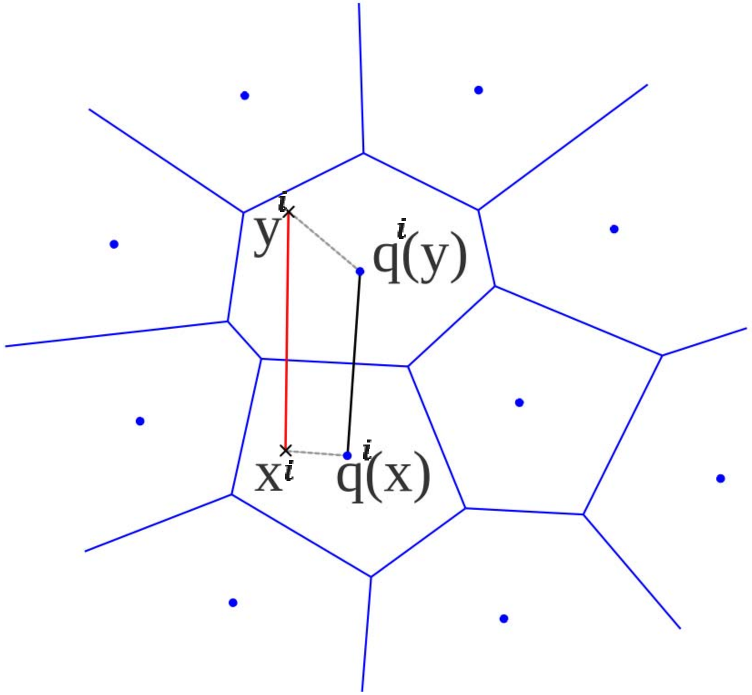
Jae-Pil Heo, Zhe Lin, and Sung-Eui Yoon

CVPR 2014

PQ: Product Quantization [Jegou et al., TPAMI 2011]

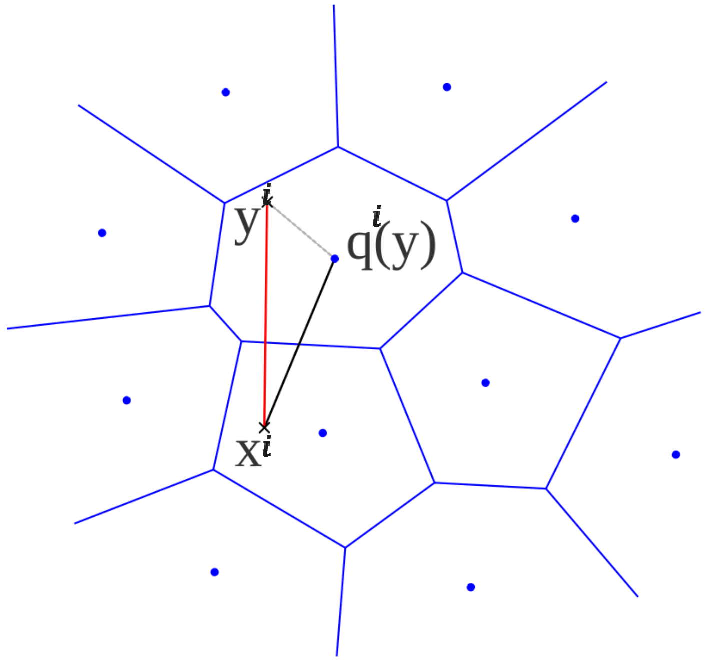


Distance Computation in PQ



Symmetric Distance

$$d_{SD}^{PQ}(x,y)^2 = \sum_{i=1}^{M} \left\| q^i(x^i) - q^i(y^i) \right\|^2$$



Asymmetric Distance

$$d_{AD}^{PQ}(x,y)^2 = \sum_{i=1}^{M} \left\| x^i - q^i(y^i) \right\|^2$$

Terms

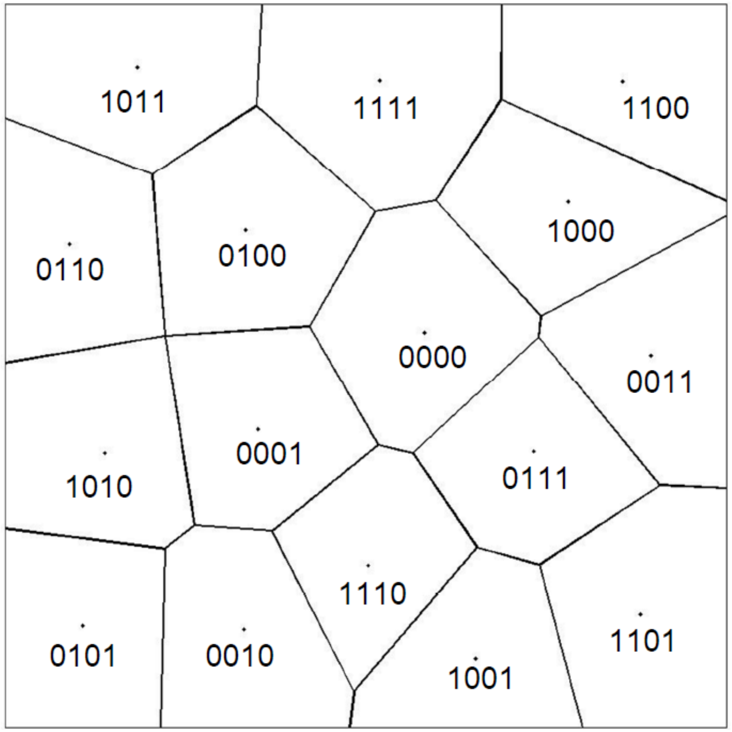
x: query, y: data, M: # of Subspaces,

q^i: quantizer in i^{th} subspace, x^i: sub-vector of x in i^{th} subspace

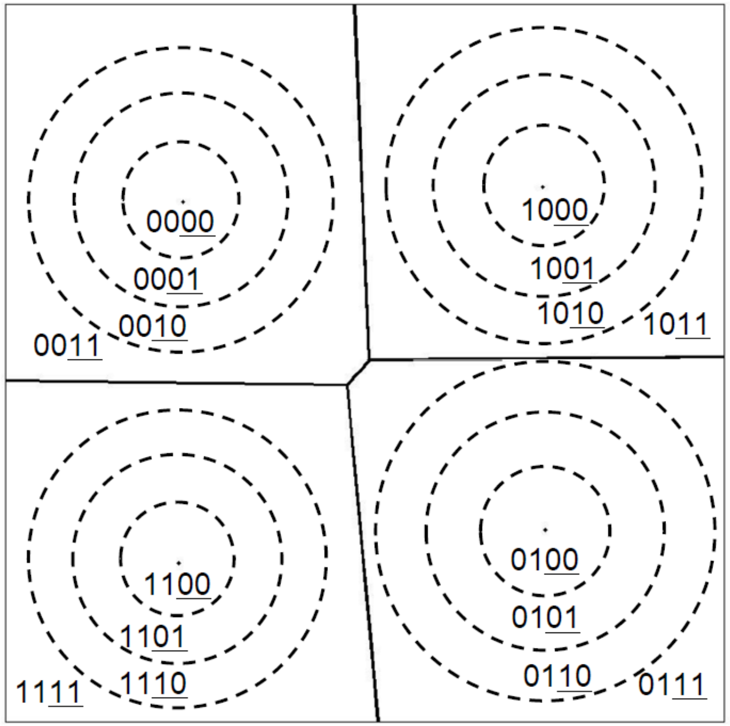
Figures are from [Jegou et al., TPAMI 2011]

DPQ: Distance Encoded PQ

- DPQ encodes quantized distance from the center as well as the cluster index in each subspace.

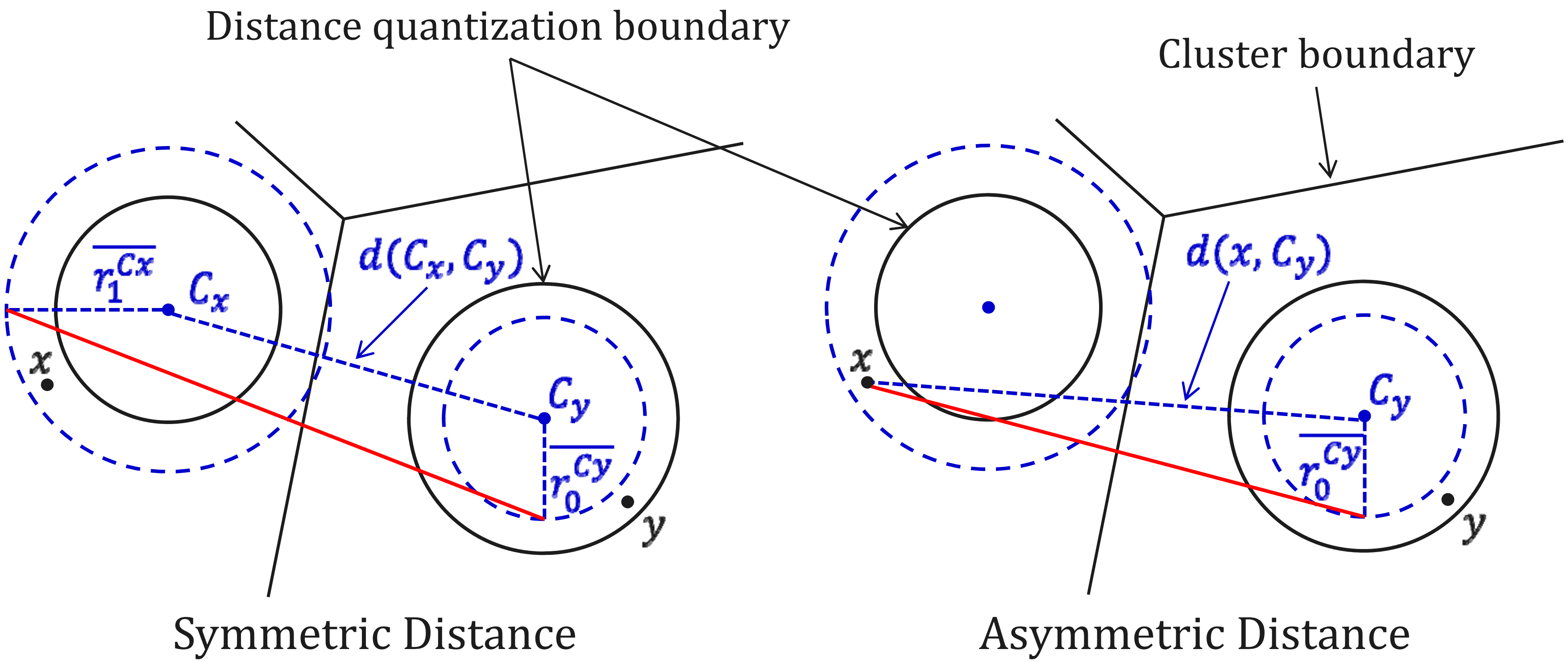


PQ example



DPQ example

Distance Computation in DPQ

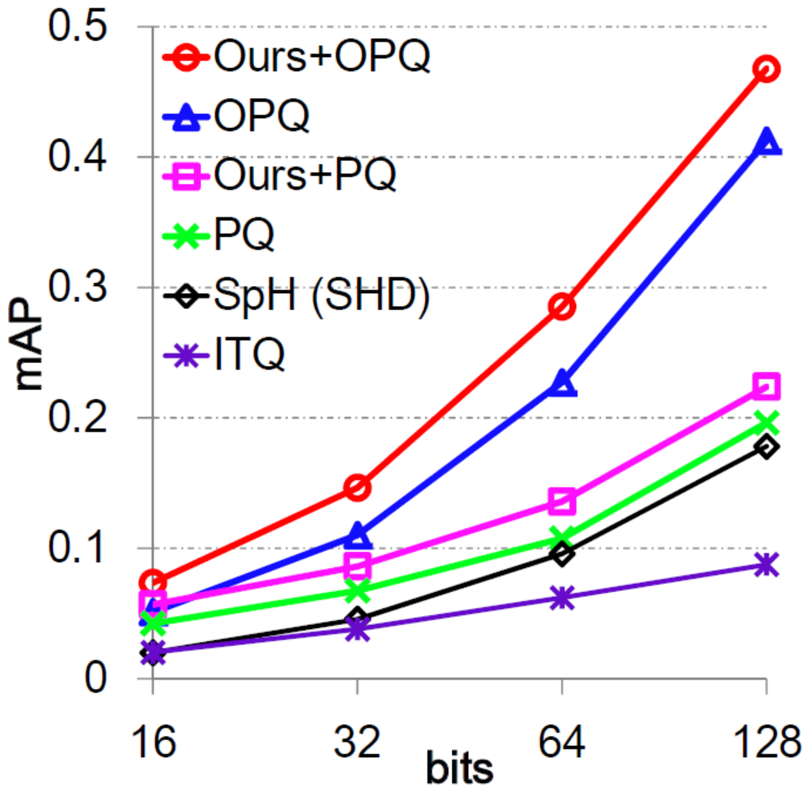


$$d_{SD}^{DPQ}(x,y)^2 = d(C_x,C_y)^2 + \overline{r_1^{Cx}}^2 + \overline{r_0^{Cy}}^2$$

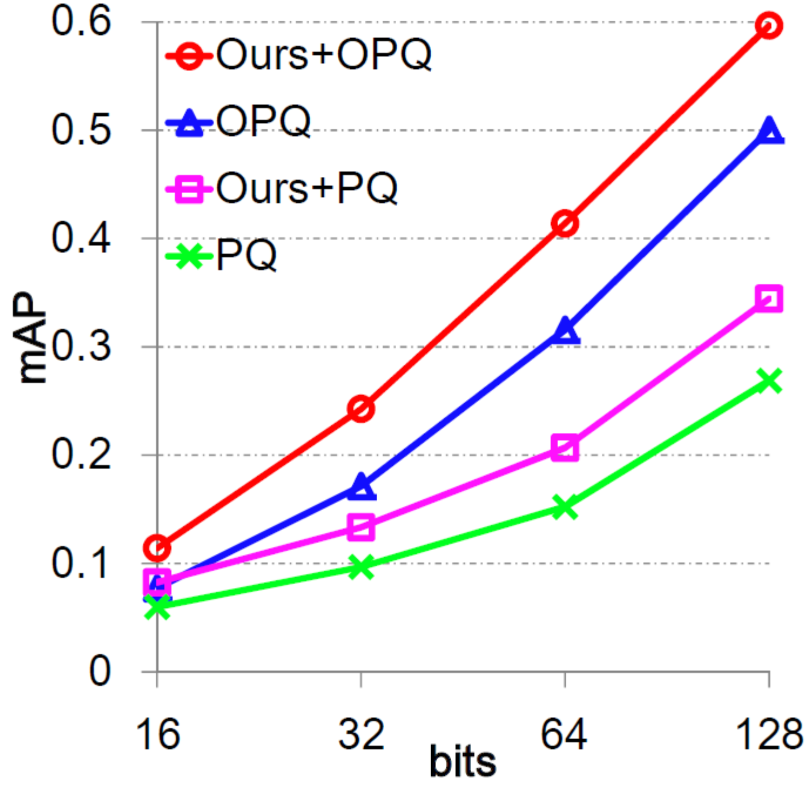
$$d_{SD}^{DPQ}(x,y)^2 = d(x,C_y)^2 + \overline{r_0^{Cy}}^2$$

$\overline{r_j^C}$: average distance from the center to points whose cluster center is C and quantized distance index is j

Results on GIST-1M-960D



Symmetric distance



Asymmetric distance

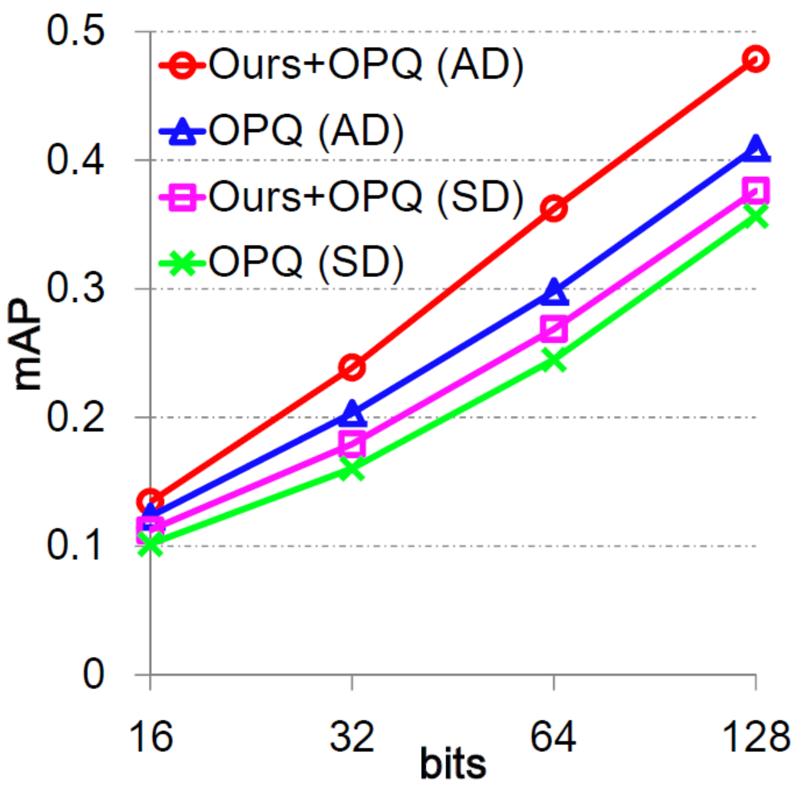
1000-nearest neighbor search mAP

OPQ: Optimized PQ [Ge et al., CVPR 2013]

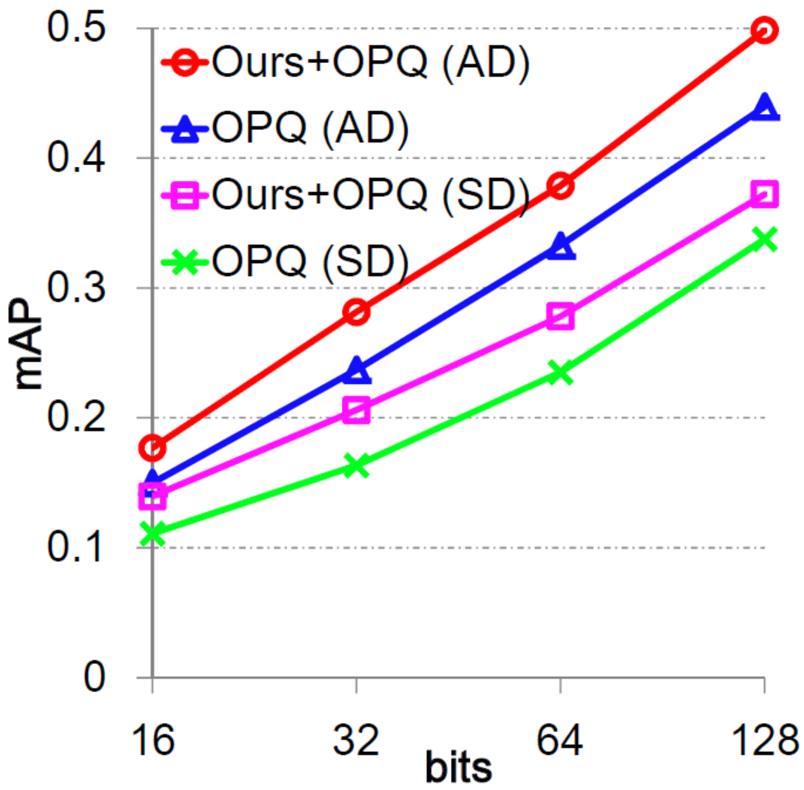
SpH: Sperical Hashing [Heo et al., CVPR 2012]

ITQ: Iterative Quantization [Gong and Lazebnik, CVPR 2011]

Results on BoW-1M-1024D



Original Data



L_2 Normalized data

1000-nearest neighbor search mAP

SD: Symmetric distance

AD: Asymmetric distance

Class Objectives were:

- **Understand the basic hashing techniques based on hyperplanes**
- **Get to know a recent one based on hyperspheres**
- **Codes are available**

http://sglab.kaist.ac.kr/software.htm

Next Time…

- Novel applications