Approximate nearest neighbor search: binary codes and vector quantization

Yannis Avrithis

University of Athens

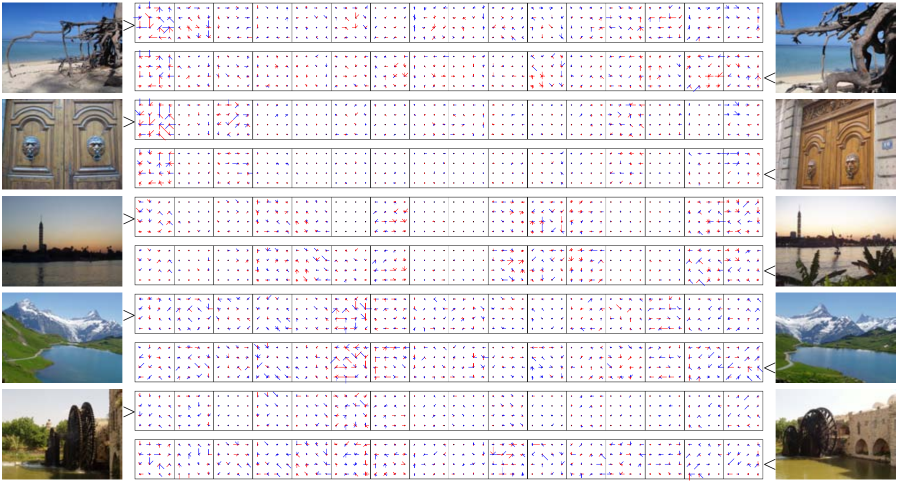
Athens, March 2015

Problem

- Given query point $\mathbf{q}$, find its nearest neighbor with respect to Euclidean distance within data set $\mathcal{X}$ in a d-dimensional space
- Focus on large scale: encode (compress) vectors, speed up distance computations
- Fit underlying distribution with little space & time overhead

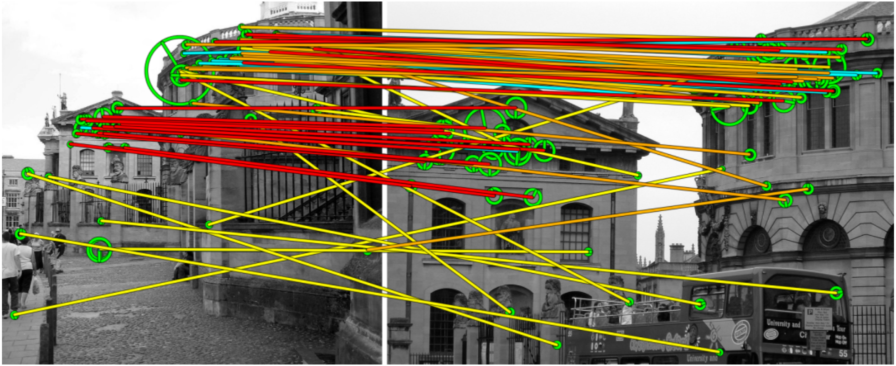
Applications in vision

Retrieval (image as point) [Jégou et al. '10][Perronnin et al. '10]



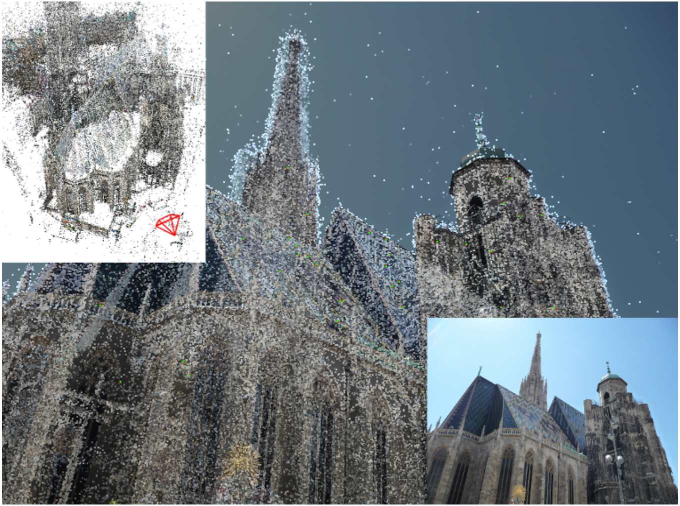
Applications in vision

Retrieval (patch as point) [Tolias et al. '13][Qin et al. '13]



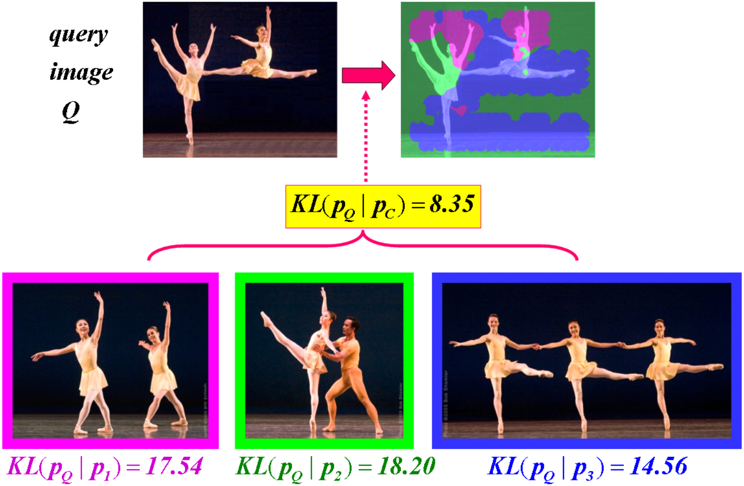
Applications in vision

Localization, pose estimation [Sattler et al. '12][Li et al. '12]



Applications in vision

Classification [Boiman et al. '08][McCann & Lowe '12]



query
image
Q

$KL(p_Q \mid p_C) = 8.35$

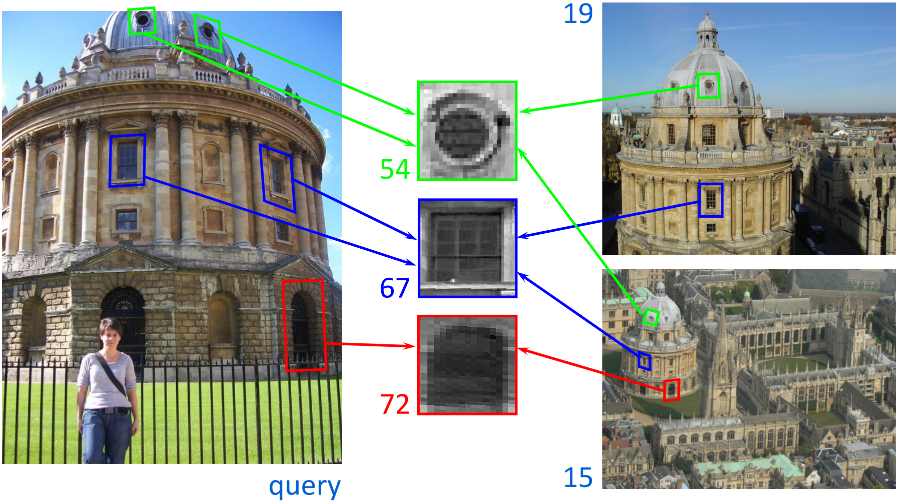
$KL(p_Q \mid p_1) = 17.54$

$KL(p_Q \mid p_2) = 18.20$

$KL(p_Q \mid p_3) = 14.56$

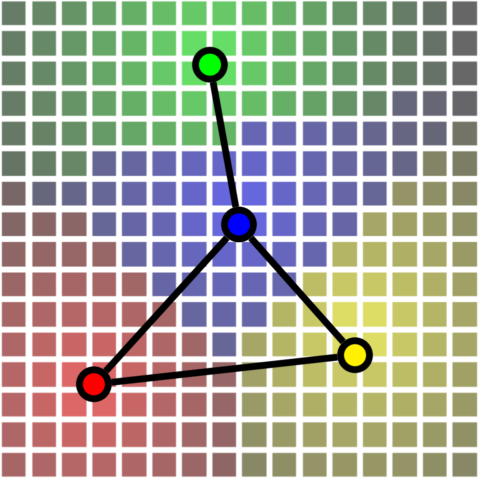
Applications in vision

Quantization [Sivic et al. '03][Philbin et al. '07]



Applications in vision

Clustering [Philbin et al. '07][Avrithis '13]



Overview

- Binary codes
  - Locality sensitive hashing [Charikar '02]
  - Spectral hashing [Weiss *et al*. '08]
  - Iterative quantization [Gong and Lazebnik '11]
- Quantization
  - Vector quantization (VQ)
  - Product quantization (PQ) [Jégou *et al*. '11]
  - Optimized product quantization (OPQ) [Ge *et al*. '13]
    Cartesian k-means [Norouzi & Fleet '13]
  - Locally optimized product quantization (LOPQ) [Kalantidis and Avrithis '14]
- Non-exhaustive search
  - Non-exhaustive PQ [Jégou *et al*. '11]
  - Inverted multi-index [Babenko & Lempitsky '12]
  - Multi-LOPQ [Kalantidis and Avrithis '14]

Overview

- Binary codes
  - Locality sensitive hashing [Charikar '02]
  - Spectral hashing [Weiss *et al*. '08]
  - Iterative quantization [Gong and Lazebnik '11]
- Quantization
  - Vector quantization (VQ)
  - Product quantization (PQ) [Jégou *et al*. '11]
  - Optimized product quantization (OPQ) [Ge *et al*. '13] Cartesian k-means [Norouzi & Fleet '13]
  - Locally optimized product quantization (LOPQ) [Kalantidis and Avrithis '14]
- Non-exhaustive search
  - Non-exhaustive PQ [Jégou *et al*. '11]
  - Inverted multi-index [Babenko & Lempitsky '12]
  - Multi-LOPQ [Kalantidis and Avrithis '14]

Overview

- Binary codes
    - Locality sensitive hashing [Charikar '02]
    - Spectral hashing [Weiss *et al*. '08]
    - Iterative quantization [Gong and Lazebnik '11]
- Quantization
    - Vector quantization (VQ)
    - Product quantization (PQ) [Jégou *et al*. '11]
    - Optimized product quantization (OPQ) [Ge *et al*. '13]
      Cartesian k-means [Norouzi & Fleet '13]
    - Locally optimized product quantization (LOPQ) [Kalantidis and Avrithis '14]
- Non-exhaustive search
    - Non-exhaustive PQ [Jégou *et al*. '11]
    - Inverted multi-index [Babenko & Lempitsky '12]
    - Multi-LOPQ [Kalantidis and Avrithis '14]

I. Binary codes

Locality sensitive hashing: random projections

[Charikar '02]

- Choose a random vector $\mathbf{a}$ from the d-dimensional Gaussian distribution.
- Define *hash function* $h_{\mathbf{a}} : \mathbb{R}^d \to \{-1, 1\}$ with

$$h_{\mathbf{a}}(\mathbf{x}) = \operatorname{sgn}(\mathbf{a} \cdot \mathbf{x}) = \begin{cases} 1, & \text{if } \mathbf{a} \cdot \mathbf{x} \geq 0 \\ -1, & \text{if } \mathbf{a} \cdot \mathbf{x} < 0. \end{cases}$$

- Then, given $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$,

$$\mathbb{P}[h_{\mathbf{a}}(\mathbf{x}) = h_{\mathbf{a}}(\mathbf{y})] = 1 - \frac{\theta(\mathbf{x}, \mathbf{y})}{\pi}$$

where $\theta(\mathbf{x}, \mathbf{y})$ is the angle between $\mathbf{x}, \mathbf{y}$.

Locality sensitive hashing: random projections

[Charikar '02]

- Choose a random vector $\mathbf{a}$ from the d-dimensional Gaussian distribution.
- Define *hash function* $h_{\mathbf{a}} : \mathbb{R}^d \to \{-1, 1\}$ with

$$h_{\mathbf{a}}(\mathbf{x}) = \operatorname{sgn}(\mathbf{a} \cdot \mathbf{x}) = \begin{cases} 1, & \text{if } \mathbf{a} \cdot \mathbf{x} \geq 0 \\ -1, & \text{if } \mathbf{a} \cdot \mathbf{x} < 0. \end{cases}$$

- Then, given $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$,

$$\mathbb{P}[h_{\mathbf{a}}(\mathbf{x}) = h_{\mathbf{a}}(\mathbf{y})] = 1 - \frac{\theta(\mathbf{x}, \mathbf{y})}{\pi}$$

where $\theta(\mathbf{x}, \mathbf{y})$ is the angle between $\mathbf{x}, \mathbf{y}$.

Binary codes and Hamming distance

- Given a set of n data points $\mathbf{x}_i \in \mathbb{R}^d$, represented by matrix $X \in \mathbb{R}^{d \times n}$.
- Define k hash functions $h_j : \mathbb{R}^d \to \{-1, 1\}$, and let $h(\mathbf{x}) = (h_1(\mathbf{x}), \ldots, h_k(\mathbf{x}))$.
- Encode each data point $\mathbf{x}$ by binary code $\mathbf{y} = h(\mathbf{x})$, and represent all encoded points by matrix $Y \in \{-1, 1\}^{k \times n}$.
  - For instance, $Y = \mathrm{sgn}(A^\top X)$ for random projections, where $A \in \mathbb{R}^{d \times k}$ represents the k random vectors.
- Now, given a query $\mathbf{q}$, encode it as $h(\mathbf{q})$ and search in Y by Hamming distance.

Binary codes and Hamming distance

- Given a set of n data points $\mathbf{x}_i \in \mathbb{R}^d$, represented by matrix $X \in \mathbb{R}^{d \times n}$.
- Define k hash functions $h_j : \mathbb{R}^d \to \{-1, 1\}$, and let $h(\mathbf{x}) = (h_1(\mathbf{x}), \dots, h_k(\mathbf{x}))$.
- Encode each data point $\mathbf{x}$ by binary code $\mathbf{y} = h(\mathbf{x})$, and represent all encoded points by matrix $Y \in \{-1, 1\}^{k \times n}$.
    - For instance, $Y = \operatorname{sgn}(A^\top X)$ for random projections, where $A \in \mathbb{R}^{d \times k}$ represents the k random vectors.
- Now, given a query $\mathbf{q}$, encode it as $h(\mathbf{q})$ and search in Y by Hamming distance.

Spectral hashing

[Weiss et al. '08]

- Define similarity matrix S with $S_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/t^2)$.
- Require binary codes to be similarity preserving, balanced, and uncorrelated:

$$\begin{array}{rl} \text{minimize} & \sum_{ij} S_{ij}\|\mathbf{y}_i - \mathbf{y}_j\|^2 \\ \text{subject to} & \mathbf{y}_i \in \{-1,1\}^k \\ & \sum_i \mathbf{y}_i = 0 \\ & \frac{1}{n}\sum_i \mathbf{y}_i\mathbf{y}_i^\top = I. \end{array}$$

Spectral hashing

[Weiss et al. ’08]

- Define similarity matrix S with $S_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/t^2)$.
- Require binary codes to be similarity preserving, balanced, and uncorrelated:

$$\begin{array}{rl} \text{minimize} & \sum_{ij} S_{ij}\|\mathbf{y}_i - \mathbf{y}_j\|^2 \\ \text{subject to} & \mathbf{y}_i \in \{-1,1\}^k \\ & \sum_i \mathbf{y}_i = 0 \\ & \frac{1}{n}\sum_i \mathbf{y}_i\mathbf{y}_i^\top = I. \end{array}$$

Spectral hashing

Relaxation

- Define Laplacian matrix $L = D - S$ with $D = \operatorname{diag}(S\mathbf{1})$.
- Problem is relaxed as

$$\begin{array}{rl} \text{minimize} & \operatorname{tr}(YLY^\top) \\ \text{subject to} & Y\mathbf{1} = 0 \\ & YY^\top = I, \end{array}$$

  and solutions are the k eigenvectors of L with minimal eigenvalue, excluding eigenvector $\mathbf{1}$ with eigenvalue 0.
- See also Laplacian eigenmaps [Belkin & Niyogi '01].

Spectral hashing

Relaxation

- Define Laplacian matrix $L = D - S$ with $D = \mathrm{diag}(S\mathbf{1})$.
- Problem is relaxed as

$$\begin{aligned} \text{minimize} \quad & \mathrm{tr}(YLY^\top) \\ \text{subject to} \quad & Y\mathbf{1} = 0 \\ & YY^\top = I, \end{aligned}$$

  and solutions are the k eigenvectors of L with minimal eigenvalue, excluding eigenvector $\mathbf{1}$ with eigenvalue 0.
- See also Laplacian eigenmaps [Belkin & Niyogi '01].

Spectral hashing

Out of sample extension

- Replace data points by probability distribution p; and Laplacian matrix by Laplacian operator L_p acting on functions.
- Then, solutions are the k eigenfunctions f of L_p (such that $L_p f = \lambda f$) with minimal eigenvalue, excluding eigenfunction $f(\mathbf{x}) = 1$ with eigenvalue 0.
- If p is uniform, then eigenfunctions have outer product form, and for 1-dimensional distribution on $[a, b]$,

$$
\begin{aligned}
\phi_j(x) &= \sin\left(\frac{\pi}{2} + \frac{j\pi}{b-a}x\right) \\
\lambda_j &= 1 - e^{-\frac{t^2}{2}\left(\frac{j\pi}{b-a}\right)^2}
\end{aligned}
$$

Spectral hashing

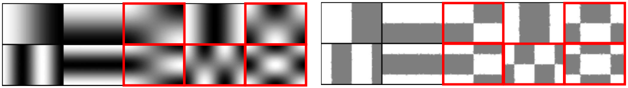
Out of sample extension

- Replace data points by probability distribution p; and Laplacian matrix by Laplacian operator L_p acting on functions.
- Then, solutions are the k eigenfunctions f of L_p (such that $L_p f = \lambda f$) with minimal eigenvalue, excluding eigenfunction $f(\mathbf{x}) = 1$ with eigenvalue 0.
- If p is uniform, then eigenfunctions have outer product form, and for 1-dimensional distribution on $[a, b]$,

$$\begin{aligned} \phi_j(x) &= \sin\left(\frac{\pi}{2} + \frac{j\pi}{b-a}x\right) \\ \lambda_j &= 1 - e^{-\frac{t^2}{2}\left(\frac{j\pi}{b-a}\right)^2} \end{aligned}$$

Spectral hashing

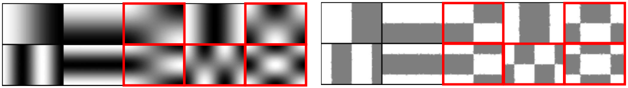
Example



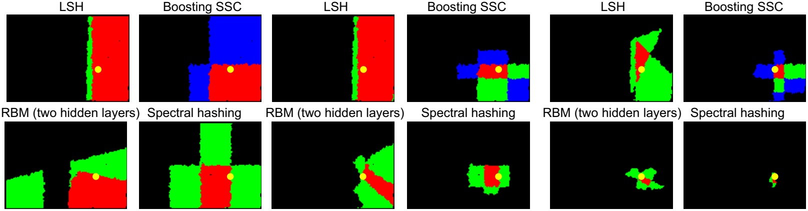
- Red: outer-product eigenfunctions: excluded
- Better to cut long dimension first
- Lower spatial frequencies are better than higher ones

Spectral hashing

Example



- Red: outer-product eigenfunctions: excluded
- Better to cut long dimension first
- Lower spatial frequencies are better than higher ones



a) 3 bits b) 7 bits c) 15 bits

- Red: radius $= 0$; green: radius $= 1$; blue: radius $= 2$

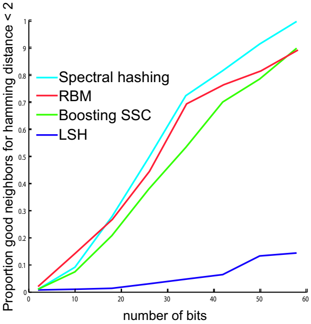
Spectral hashing

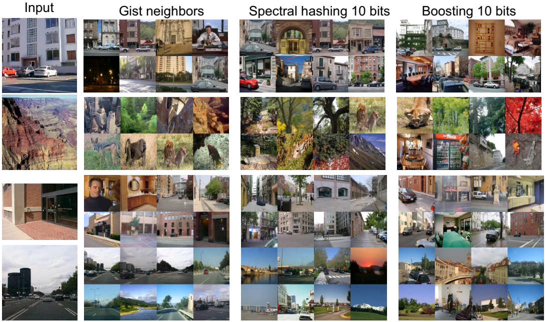
Algorithm

1. Rotate data points by PCA.
2. Evaluate k smallest eigenvalues for each PCA direction.
3. Sort the kd eigenvalues, exclude outer-product ones, and select the k smallest.
4. Set hash function $h_j(\mathbf{x}) = \operatorname{sgn}(\phi_j(\mathbf{x}))$ for each of the corresponding k eigenfunctions ϕ_j.

Spectral hashing

Result on LabelMe

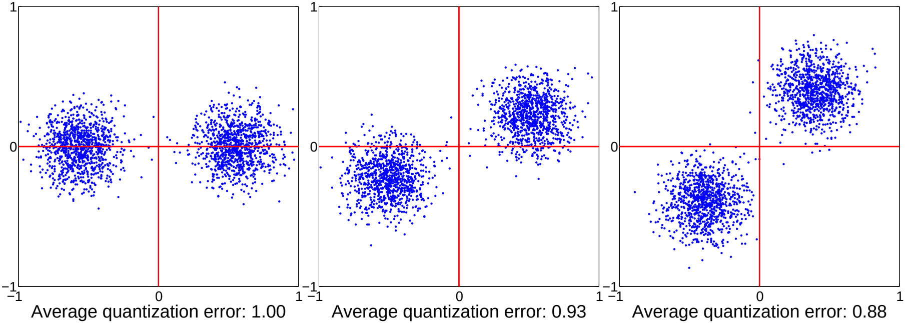




Iterative quantization

[Gong and Lazebnik '11]

Quantize each data point to the closest vertex of the binary cube, $(\pm 1, \pm 1)$.



(a) PCA aligned. (b) Random Rotation. (c) Optimized Rotation.

Iterative quantization

Formulation

- Assume data points to be zero centered, $X\mathbf{1} = 0$.
- Assume hash functions $y^j = h_j(\mathbf{x}) = \mathrm{sgn}(\mathbf{a}_j \cdot \mathbf{x})$, or $Y = \mathrm{sgn}(A^\top X)$.
- Drop similarity preservation
- Balance $h_j(\mathbf{x}) \cdot \mathbf{1} = 0$ is equivalent to variance of $h_j(\mathbf{x})$ being maximized:

$$\begin{array}{rl} \text{maximize} & \sum_j \mathrm{var}(\mathrm{sgn}(\mathbf{a}_j^\top X)) \\ \text{subject to} & \frac{1}{n} Y Y^\top = I. \end{array}$$

Iterative quantization

Relaxation

- Drop sgn.
- Relax correlation constraint by just requiring hyperplanes to be orthogonal:

$$\begin{array}{rl} \text{maximize} & \mathrm{tr}(A^\top X X^\top A) \\ \text{subject to} & A^\top A = I, \end{array}$$

  and a solution consists of the k eigenvectors of data covariance matrix $XX^\top$ with maximal eigenvalue.

- See also semi-supervised hashing [Wang *et al*. '10].

Iterative quantization

Relaxation

- Drop sgn.
- Relax correlation constraint by just requiring hyperplanes to be orthogonal:

$$\begin{array}{rl} \text{maximize} & \mathrm{tr}(A^\top X X^\top A) \\ \text{subject to} & A^\top A = I, \end{array}$$

  and a solution consists of the k eigenvectors of data covariance matrix $XX^\top$ with maximal eigenvalue.
- See also semi-supervised hashing [Wang *et al.* '10].

Iterative quantization

Refinement

- But, if A is an optimal solution, then so is $AR^\top$ for orthogonal $R \in \mathbb{R}^{k\times k}$.
- So, if $Z = A^\top X$ is the projected data, define loss

$$E(Y, R) = \|Y - RZ\|_F^2$$

  and repeat
  - Fix R, update $Y \leftarrow \mathrm{sgn}(RZ)$
  - Fix Y, update $R \leftarrow UV^\top$ where $YZ^\top = USV^\top$ (align by SVD)
- See also multiclass spectral clustering [Yu & Shi '03].

Iterative quantization

Refinement

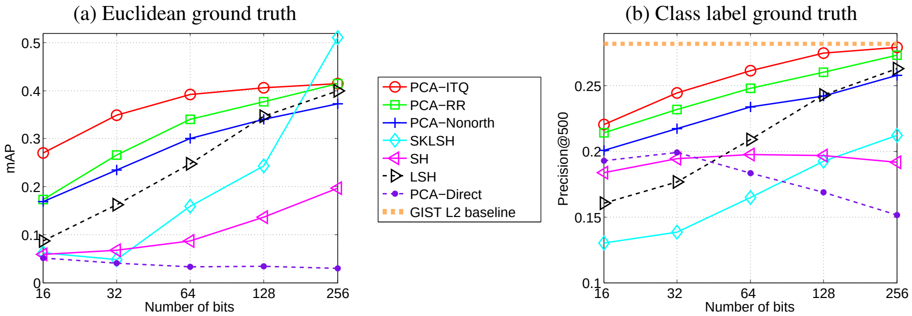
- But, if A is an optimal solution, then so is $AR^\top$ for orthogonal $R \in \mathbb{R}^{k\times k}$.
- So, if $Z = A^\top X$ is the projected data, define loss

  $$E(Y, R) = \|Y - RZ\|_F^2$$

  and repeat
  - Fix R, update $Y \leftarrow \mathrm{sgn}(RZ)$
  - Fix Y, update $R \leftarrow UV^\top$ where $YZ^\top = USV^\top$ (align by SVD)
- See also multiclass spectral clustering [Yu & Shi '03].

Iterative quantization

Result on CIFAR

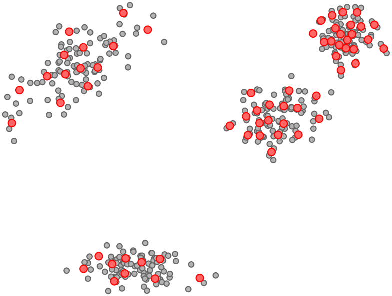

(a) Euclidean ground truth

(b) Class label ground truth

II. Quantization

Vector quantization

[Gray '84]



$$\text{minimize } E(\mathcal{C}) = \sum_{\mathbf{x}\in\mathcal{X}} \min_{\mathbf{c}\in\mathcal{C}} \|\mathbf{x}-\mathbf{c}\|^2 = \sum_{\mathbf{x}\in\mathcal{X}} \|\mathbf{x}-q(\mathbf{x})\|^2$$

distortion

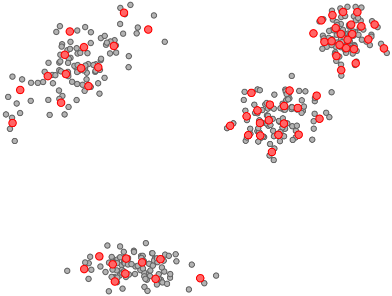
dataset

codebook

quantizer

Vector quantization

[Gray '84]



$$\text{minimize } E(\mathcal{C}) = \sum_{\mathbf{x}\in\mathcal{X}} \min_{\mathbf{c}\in\mathcal{C}} \|\mathbf{x}-\mathbf{c}\|^2 = \sum_{\mathbf{x}\in\mathcal{X}} \|\mathbf{x}-q(\mathbf{x})\|^2$$

distortion

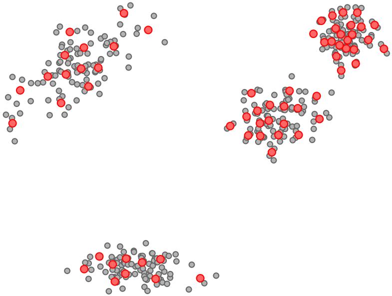
dataset

codebook

quantizer

Vector quantization

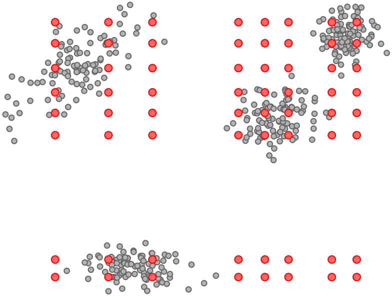
[Gray '84]



- For small distortion → large $k = |\mathcal{C}|$:
  - hard to train
  - too large to store
  - too slow to search

Product quantization

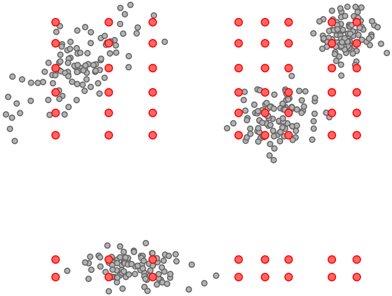
[Jégou et al. '11]



$$\begin{aligned}\text{minimize} \quad & \sum_{\mathbf{x}\in\mathcal{X}} \min_{\mathbf{c}\in\mathcal{C}} \|\mathbf{x}-\mathbf{c}\|^2 \\ \text{subject to} \quad & \mathcal{C} = \mathcal{C}^1 \times \cdots \times \mathcal{C}^m\end{aligned}$$

Product quantization

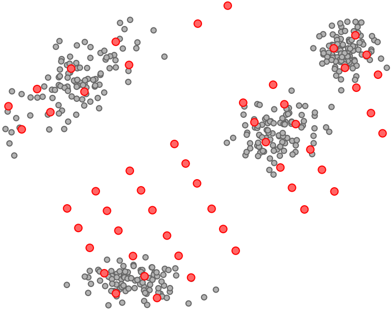
[Jégou et al. '11]



- train: $q = (q^1, \dots, q^m)$ where $q^1, \dots, q^m$ obtained by VQ
- store: $|\mathcal{C}| = k^m$ with $|\mathcal{C}^1| = \cdots = |\mathcal{C}^m| = k$
- search: $\|\mathbf{y} - q(\mathbf{x})\|^2 = \sum_{j=1}^{m} \|\mathbf{y}^j - q^j(\mathbf{x}^j)\|^2$ where $q^j(\mathbf{x}^j) \in \mathcal{C}^j$

Optimized product quantization

[Ge et al. '13]



$$\begin{aligned} &\text{minimize} \quad \sum_{\mathbf{x}\in\mathcal{X}} \min_{\hat{\mathbf{c}}\in\hat{\mathcal{C}}} \|\mathbf{x} - R\hat{\mathbf{c}}\|^2 \\ &\text{subject to} \quad \hat{\mathcal{C}} = \mathcal{C}^1 \times \cdots \times \mathcal{C}^m \\ &\qquad\qquad\quad\; R^\top R = I \end{aligned}$$

Optimized product quantization

Non-parametric solution

$$\begin{array}{ll} \text{rotate:} & \hat{X} \leftarrow RX \\ \text{update:} & q \leftarrow \mathsf{PQ}(\hat{X}) \quad \text{[one step]} \\ \text{assign:} & Y \leftarrow q(\hat{X}) \\ \text{align:} & R \leftarrow UV^\top \text{ where } YX^\top = USV^\top \end{array}$$

- From PQ only one step of centroid update is needed, because update of R does not alter assignment.
- Alignment minimizes $\|Y - RX\|_F^2$, as in ITQ.

Optimized product quantization

Non-parametric solution

$$\begin{array}{ll} \text{rotate:} & \hat{X} \leftarrow RX \\ \text{update:} & q \leftarrow \mathsf{PQ}(\hat{X}) \quad \text{[one step]} \\ \text{assign:} & Y \leftarrow q(\hat{X}) \\ \text{align:} & R \leftarrow UV^\top \text{ where } YX^\top = USV^\top \end{array}$$

- From PQ only one step of centroid update is needed, because update of R does not alter assignment.
- Alignment minimizes $\|Y - RX\|_F^2$, as in ITQ.

Optimized product quantization

Parametric solution for $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \Sigma)$

- From rate-distortion theory, distortion satisfies

$$E \geq k^{-2/d} d |\Sigma|^{1/d}$$

and practical distortion achieved by k-means is typically within $\sim 5\%$ of the bound. So after rotation $\hat{\Sigma} = R \Sigma R^\top$,

$$E_{\mathsf{PQ}} \geq k^{-2m/d} \frac{d}{m} \sum_{i=1}^{m} |\hat{\Sigma}_{ii}|^{m/d}$$

- But, by *arithmetic-geometric means* and *Fisher's* inequalities,

$$\frac{1}{m} \sum_{i=1}^{m} |\hat{\Sigma}_{ii}|^{m/d} \geq \prod_{i=1}^{m} |\hat{\Sigma}_{ii}|^{1/d} \geq |\hat{\Sigma}|^{1/d} = |\Sigma|^{1/d}$$

with equality implying balanced variance and independence.

Optimized product quantization

Parametric solution for $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \Sigma)$

- From rate-distortion theory, distortion satisfies

$$E \geq k^{-2/d} d |\Sigma|^{1/d}$$

  and practical distortion achieved by k-means is typically within $\sim 5\%$ of the bound. So after rotation $\hat{\Sigma} = R \Sigma R^{\top}$,

$$E_{\mathsf{PQ}} \geq k^{-2m/d} \frac{d}{m} \sum_{i=1}^{m} |\hat{\Sigma}_{ii}|^{m/d}$$

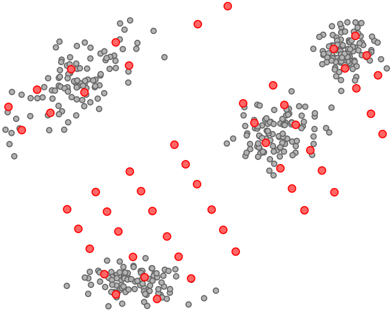
- But, by *arithmetic-geometric means* and *Fisher's* inequalities,

$$\frac{1}{m} \sum_{i=1}^{m} |\hat{\Sigma}_{ii}|^{m/d} \geq \prod_{i=1}^{m} |\hat{\Sigma}_{ii}|^{1/d} \geq |\hat{\Sigma}|^{1/d} = |\Sigma|^{1/d}$$

  with equality implying balanced variance and independence.

Optimized product quantization

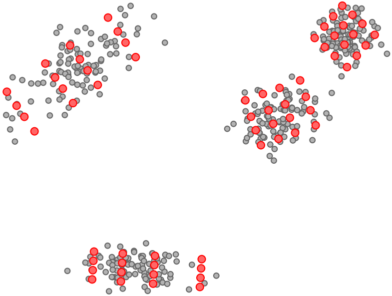
Parametric solution for $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \Sigma)$



- independence: PCA-align by diagonalizing Σ as $U \Lambda U^\top$
- balanced variance: permute Λ by π such that $\prod_i \lambda_i$ is constant in each subspace; $R \leftarrow U P_\pi^\top$
- find $\hat{\mathcal{C}}$ by PQ on rotated data $\hat{X} = RX$

Locally optimized product quantization

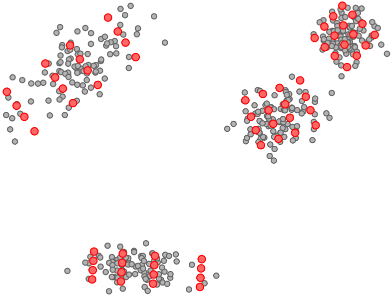
[Kalantidis & Avrithis '14]



- compute residuals $r(\mathbf{x}) = \mathbf{x} - q(\mathbf{x})$ on coarse quantizer q
- collect residuals $\mathcal{Z}_i = \{r(\mathbf{x}) : q(\mathbf{x}) = \mathbf{c}_i\}$ per cell
- train $(R_i, q_i) \leftarrow \text{OPQ}(\mathcal{Z}_i)$ per cell

Locally optimized product quantization

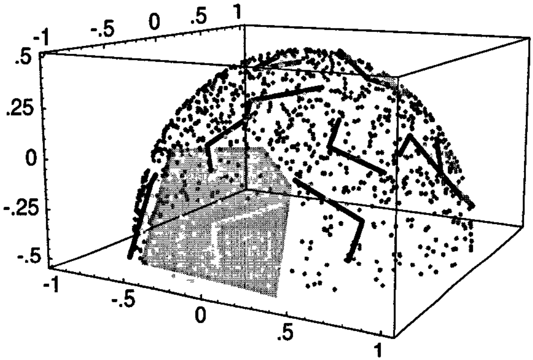
[Kalantidis & Avrithis '14]



- residual distributions closer to Gaussian assumption
- better captures the support of data distribution, like local PCA
    - multimodal (e.g. mixture) distributions
    - distributions on nonlinear manifolds

Local principal component analysis

[Kambhatla & Leen '97]

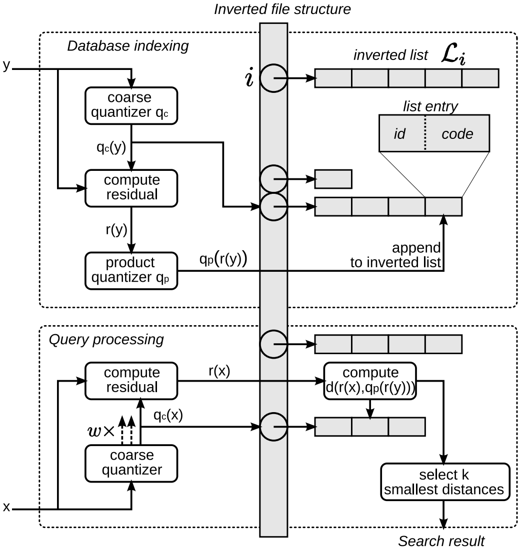


But, we are not doing dimensionality reduction!

III. Non-exhaustive search

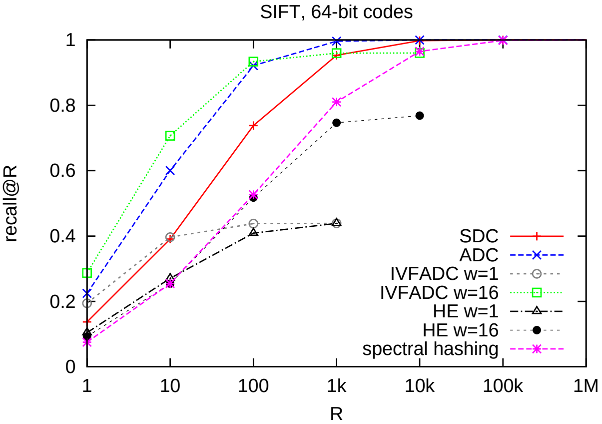
Non-exhaustive search

[Jégou et al. ’11]



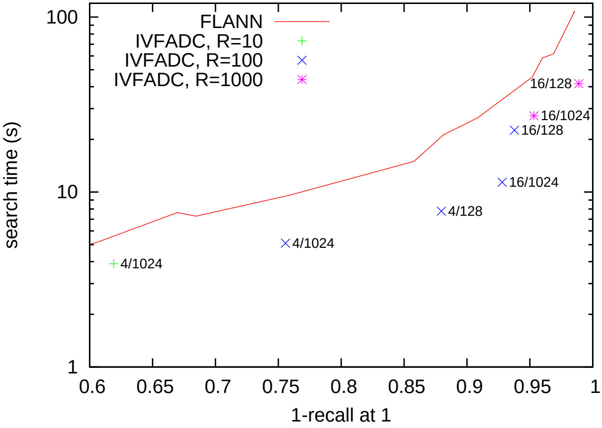
Product quantization

Result on SIFT1M



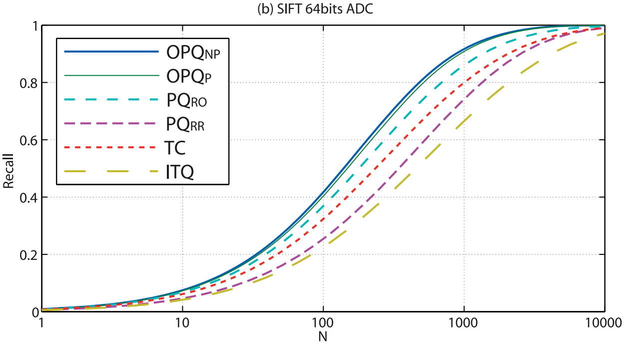
Product quantization

vs. FLANN on SIFT1M



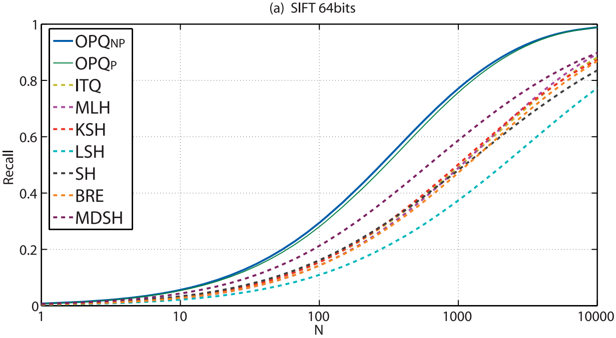
Optimized product quantization

Result on SIFT1M



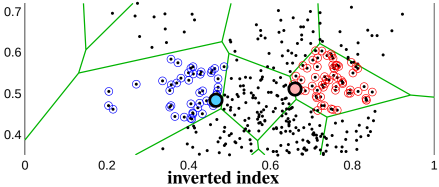
Optimized product quantization

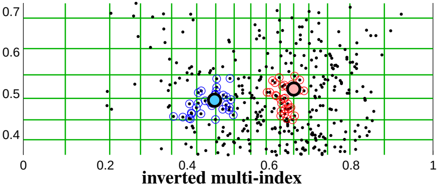
vs. binary codes on SIFT1M



Inverted multi-index

[Babenko & Lempitsky ’12]

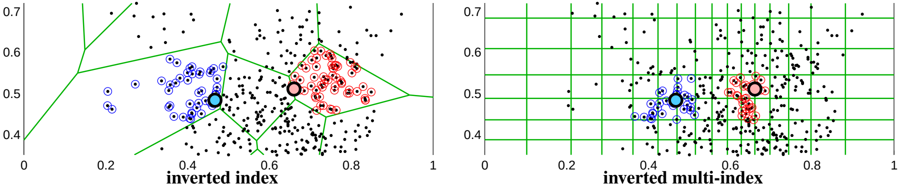




- decompose vectors as $\mathbf{x} = (\mathbf{x}^1, \mathbf{x}^2)$
- train codebooks $\mathcal{C}^1, \mathcal{C}^2$ from datasets $\{\mathbf{x}_n^1\}, \{\mathbf{x}_n^2\}$
- induced codebook $\mathcal{C}^1 \times \mathcal{C}^2$ gives a finer partition
- given query $\mathbf{q}$, visit cells $(\mathbf{c}^1, \mathbf{c}^2) \in \mathcal{C}^1 \times \mathcal{C}^2$ in ascending order of distance to $\mathbf{q}$ by multi-sequence algorithm

Inverted multi-index

[Babenko & Lempitsky '12]

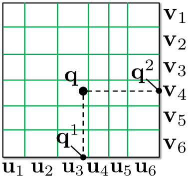


- decompose vectors as $\mathbf{x} = (\mathbf{x}^1, \mathbf{x}^2)$
- train codebooks $\mathcal{C}^1, \mathcal{C}^2$ from datasets $\{\mathbf{x}_n^1\}, \{\mathbf{x}_n^2\}$
- induced codebook $\mathcal{C}^1 \times \mathcal{C}^2$ gives a finer partition
- given query $\mathbf{q}$, visit cells $(\mathbf{c}^1, \mathbf{c}^2) \in \mathcal{C}^1 \times \mathcal{C}^2$ in ascending order of distance to $\mathbf{q}$ by multi-sequence algorithm

Inverted multi-index

Multi-sequence algorithm

space subdivision via PQ



⇨ product quantization

$\mathbf{q}^1$ vs. $\mathcal{U}$

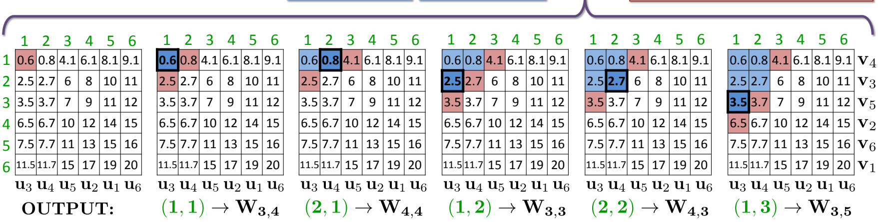
i	$\mathbf{u}_{\alpha(i)}$	r
1	$\mathbf{u_3}$	0.5
2	$\mathbf{u_4}$	0.7
3	$\mathbf{u_5}$	4
4	$\mathbf{u_2}$	6
5	$\mathbf{u_1}$	8
6	$\mathbf{u_6}$	9

$\mathbf{q}^2$ vs. $\mathcal{V}$

j	$\mathbf{v}_{\beta(j)}$	s
1	$\mathbf{v_4}$	0.1
2	$\mathbf{v_3}$	2
3	$\mathbf{v_5}$	3
4	$\mathbf{v_2}$	6
5	$\mathbf{v_6}$	7
6	$\mathbf{v_1}$	11

⇨ multi-sequence algorithm

$[\mathbf{u}_{\alpha(i)}\ \mathbf{v}_{\beta(j)}]$	(i,j)	$r(i)+s(j)$
$[\mathbf{u_3}\ \mathbf{v_4}]$	(1,1)	0.6 (0.5+0.1)
$[\mathbf{u_4}\ \mathbf{v_4}]$	(2,1)	0.8 (0.7+0.1)
$[\mathbf{u_3}\ \mathbf{v_3}]$	(1,2)	2.5 (0.5+2)
$[\mathbf{u_4}\ \mathbf{v_3}]$	(2,2)	2.7 (0.7+2)
$[\mathbf{u_3}\ \mathbf{v_5}]$	(1,3)	3.5 (0.5+3)
$[\mathbf{u_4}\ \mathbf{v_5}]$	(2,3)	3.7 (0.7+3)
$[\mathbf{u_5}\ \mathbf{v_4}]$	(3,1)	4.1 (4+0.1)
$[\mathbf{u_5}\ \mathbf{v_3}]$	(3,2)	6 (4+2)
$[\mathbf{u_3}\ \mathbf{v_2}]$	(1,4)	6.5 (0.5+6)
…		

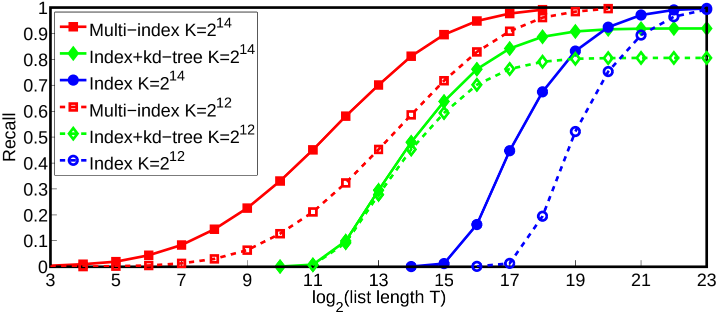


	1	2	3	4	5	6	
1	0.6	0.8	4.1	6.1	8.1	9.1	$\mathbf{v}_4$
2	2.5	2.7	6	8	10	11	$\mathbf{v}_3$
3	3.5	3.7	7	9	11	12	$\mathbf{v}_5$
4	6.5	6.7	10	12	14	15	$\mathbf{v}_2$
5	7.5	7.7	11	13	15	16	$\mathbf{v}_6$
6	11.5	11.7	15	17	19	20	$\mathbf{v}_1$
	$\mathbf{u}_3$	$\mathbf{u}_4$	$\mathbf{u}_5$	$\mathbf{u}_2$	$\mathbf{u}_1$	$\mathbf{u}_6$	

OUTPUT: $(1,1) \rightarrow \mathbf{W}_{3,4}$ $(2,1) \rightarrow \mathbf{W}_{4,4}$ $(1,2) \rightarrow \mathbf{W}_{3,3}$ $(2,2) \rightarrow \mathbf{W}_{4,3}$ $(1,3) \rightarrow \mathbf{W}_{3,5}$

Inverted multi-index

Result on SIFT1B: are NN in candidate lists?



Locally optimized product quantization

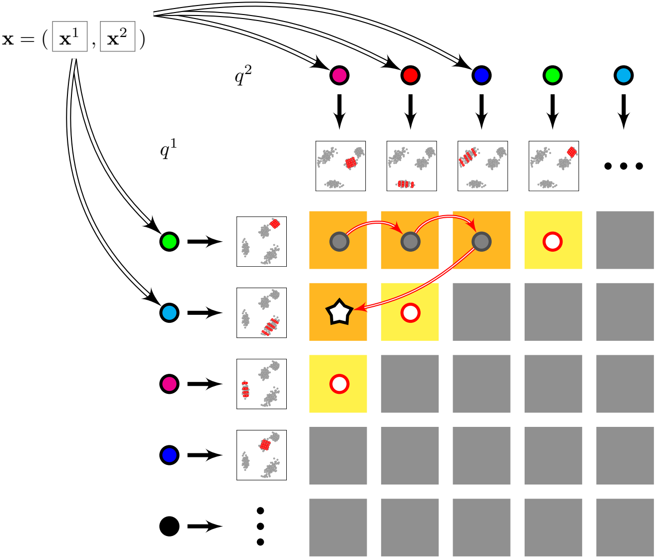
Result on SIFT1B, 64-bit codes

Method	$R = 1$	$R = 10$	$R = 100$
Ck-means [Norouzi & Fleet '13]	–	–	0.649
IVFADC [Jégou *et al.* '11]	0.106	0.379	0.748
IVFADC [Jégou *et al.* '11]	0.088	0.372	0.733
OPQ [Ge *et al.* '13]	0.114	0.399	0.777
Multi-D-ADC [Babenko & Lempitsky '12]	0.165	0.517	0.860
LOR+PQ [Kalantidis & Avrithis '14]	0.183	0.565	0.889
LOPQ [Kalantidis & Avrithis '14]	0.199	0.586	0.909

Most benefit comes from locally optimized rotation!

Multi-LOPQ

[Kalantidis & Avrithis '14]



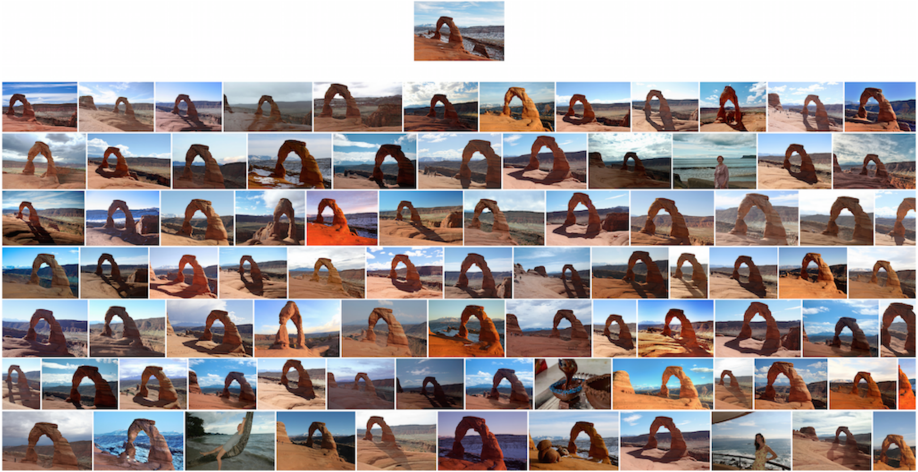
Multi-LOPQ

Result on SIFT1B, 128-bit codes

T	Method	$R = 1$	10	100
20K	IVFADC+R [Jégou *et al.* '11]	0.262	0.701	0.962
	LOPQ+R [Kalantidis & Avrithis '14]	0.350	0.820	0.978
10K	Multi-D-ADC [Babenko & Lempitsky '12]	0.304	0.665	0.740
	OMulti-D-OADC [Ge *et al.* '13]	0.345	0.725	0.794
	Multi-LOPQ [Kalantidis & Avrithis '14]	0.430	0.761	0.782
30K	Multi-D-ADC [Babenko & Lempitsky '12]	0.328	0.757	0.885
	OMulti-D-OADC [Ge *et al.* '13]	0.366	0.807	0.913
	Multi-LOPQ [Kalantidis & Avrithis '14]	0.463	0.865	0.905
100K	Multi-D-ADC [Babenko & Lempitsky '12]	0.334	0.793	0.959
	OMulti-D-OADC [Ge *et al.* '13]	0.373	0.841	0.973
	Multi-LOPQ [Kalantidis & Avrithis '14]	0.476	0.919	0.973

Multi-LOPQ

Image query on Flickr 100M (deep learned features, 4k → 128 dimensions)



`http://image.ntua.gr/iva/research/`

Thank you!