Large scale clustering and nearest neighbor search

Yannis Avrithis

Inria Rennes-Bretagne Atlantique

Prague, March 2016

Problem

ANN search

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

Vector quantization

- Given data set $\mathcal{X}$, map it to discrete codebook $\mathcal{C}$ such that distortion is minimized
- Use ANN search to assign points to centroids
- Use vector quantization to improve ANN search

Problem

ANN search

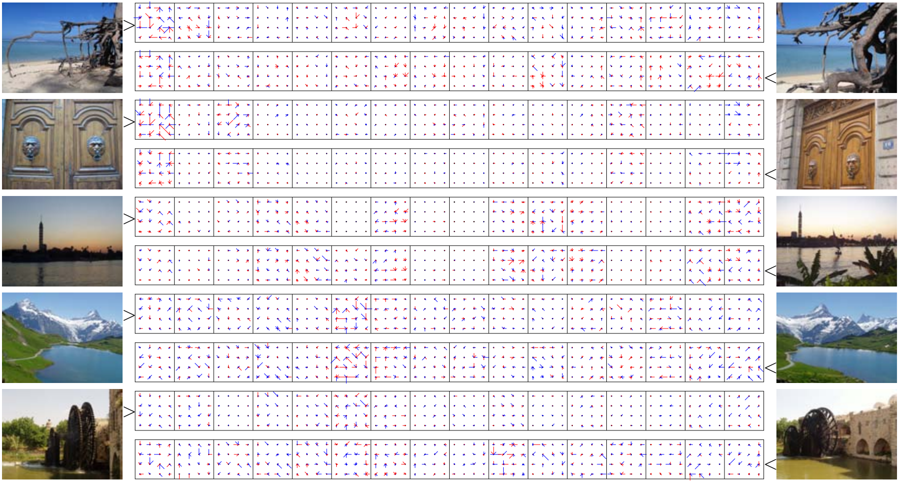
- Given query point $\mathbf{q}$, find its nearest neighbor with respect to Euclidean distance within data set $\mathcal{X}$ in a d-dimensional space
- Encode (compress) vectors, speed up distance computations
- Fit underlying distribution with little space & time overhead

Vector quantization

- Given data set $\mathcal{X}$, map it to discrete codebook $\mathcal{C}$ such that distortion is minimized
- Use ANN search to assign points to centroids
- Use vector quantization to improve ANN search

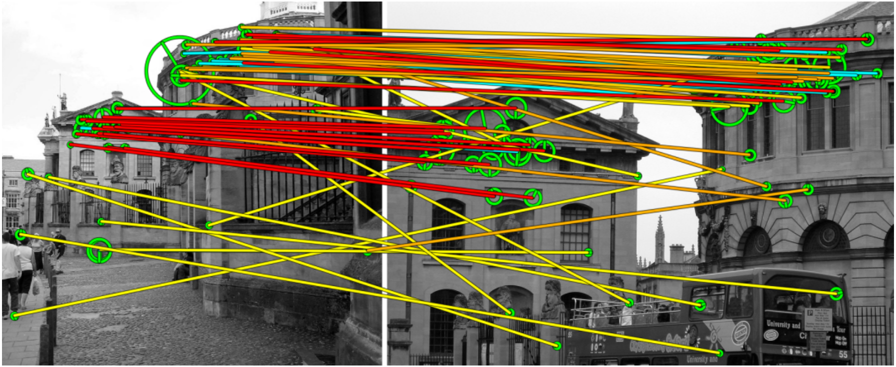
Applications in vision

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



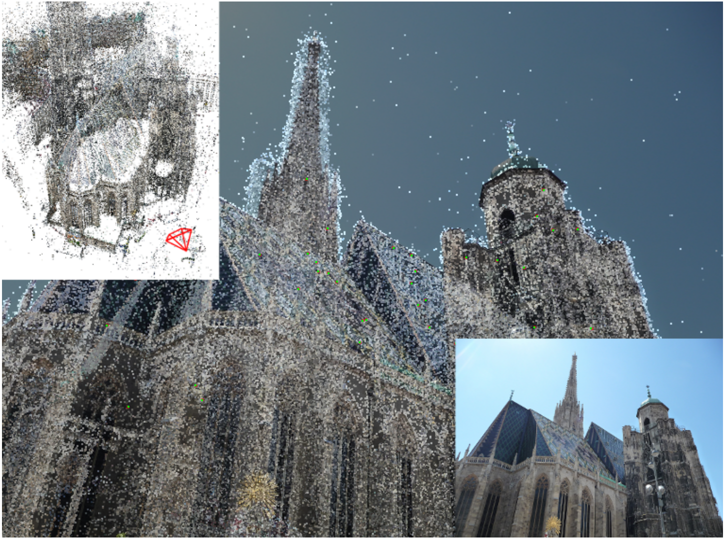
Applications in vision

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



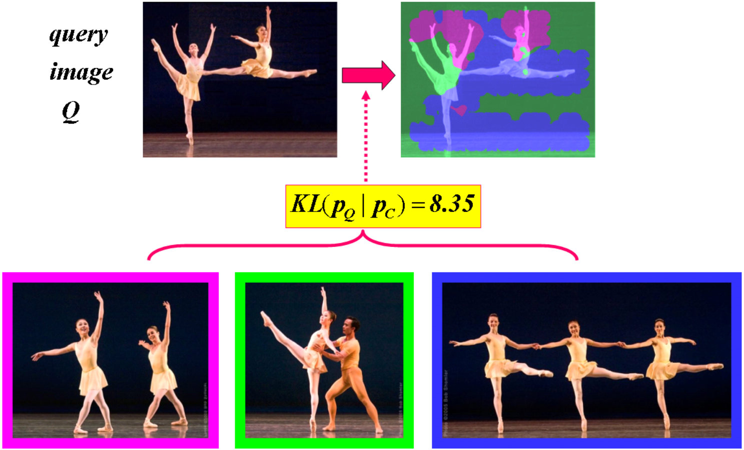
Applications in vision

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



Applications in vision

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



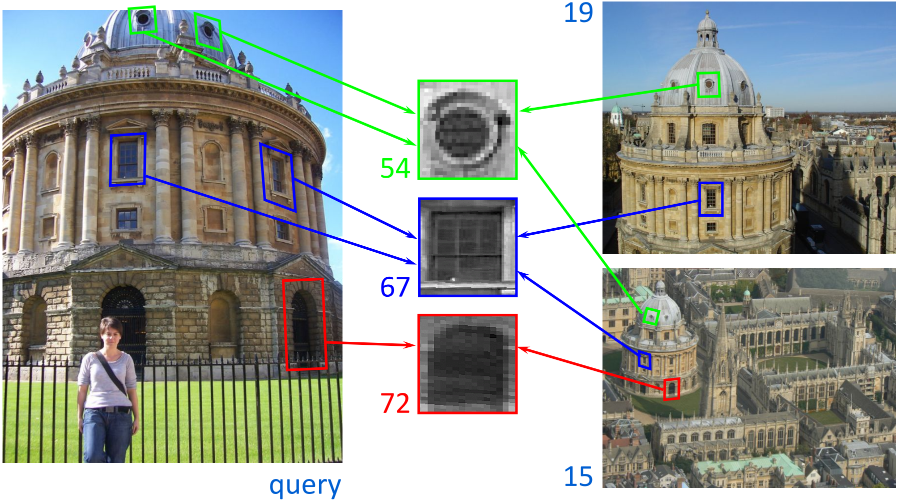
query image Q

$KL(p_Q \mid p_C) = 8.35$

$KL(p_Q \mid p_1) = 17.54$ $KL(p_Q \mid p_2) = 18.20$ $KL(p_Q \mid p_3) = 14.56$

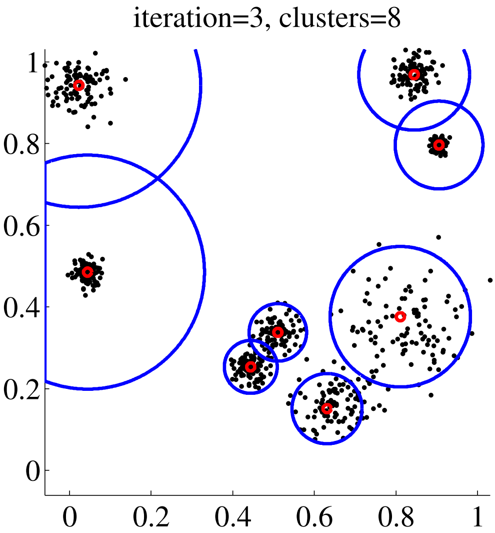
Applications in vision

BoW (patch quantization) [Sivic et al. ’03][Philbin et al. ’07]



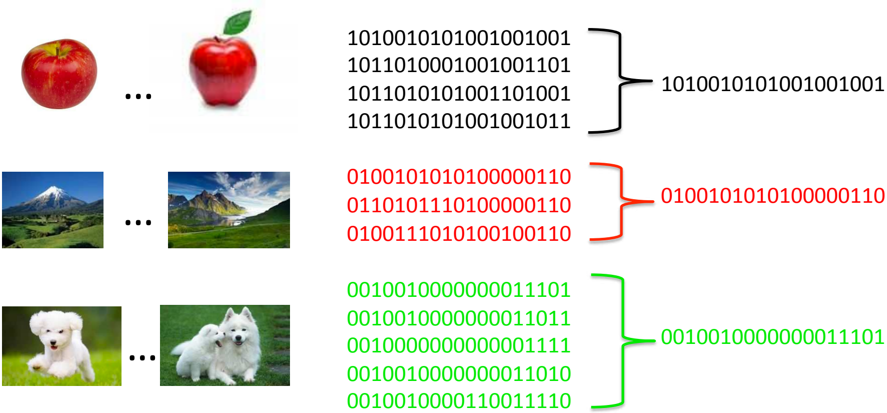
Applications in vision

BoW (codebook construction) [Philbin et al. '07][Avrithis '12]



Applications in vision

Image clustering [Gong et al. ’15][Avrithis ’15]



1010010101001001001
1011010001001001101
1011010101001101001
1011010101001001011

1010010101001001001

0100101010100000110
0110101110100000110
0100111010100100110

0100101010100000110

0010010000000011101
0010010000000011011
0010000000000001111
0010010000000011010
0010010000110011110

0010010000000011101

Images

binary hashes

binary cluster centers

Overview (1)

Binary codes

- spectral hashing [Weiss *et al.* '08]
- iterative quantization [Gong & Lazebnik '11]

Quantization

- vector quantization (VQ) [Gray '84]
- 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 & Avrithis '14]

Overview (1)

Binary codes

- spectral hashing [Weiss *et al*. '08]
- iterative quantization [Gong & Lazebnik '11]

Quantization

- vector quantization (VQ) [Gray '84]
- 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 & Avrithis '14]

Overview (2)

Non-exhaustive search

- non-exhaustive PQ [Jégou *et al.* '11]
- inverted multi-index [Babenko & Lempitsky '12]
- multi-LOPQ [Kalantidis & Avrithis '14]

Clustering

- hierarchical k-means [Nister & Stewenius '06]
- approximate k-means [Philbin *et al.* '07]
- approximate Gaussian mixtures [Kalantidis & Avrithis '12]
- dimensionality-recursive vector quantization [Avrithis '13]
- ranked retrieval [Broder *et al.* '14]
- inverted-quantized k-means [Avrithis *et al.* '15]

Overview (2)

Non-exhaustive search

- non-exhaustive PQ [Jégou *et al.* '11]
- inverted multi-index [Babenko & Lempitsky '12]
- multi-LOPQ [Kalantidis & Avrithis '14]

Clustering

- hierarchical k-means [Nister & Stewenius '06]
- approximate k-means [Philbin *et al.* '07]
- approximate Gaussian mixtures [Kalantidis & Avrithis '12]
- dimensionality-recursive vector quantization [Avrithis '13]
- ranked retrieval [Broder *et al.* '14]
- inverted-quantized k-means [Avrithis *et al.* '15]

Binary codes

Spectral hashing

[Weiss et al. '08]

- Given a set of n data points $\mathbf{x}_i \in \mathbb{R}^d$, encode each by binary code $\mathbf{y}_i$
- 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:

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

Spectral hashing

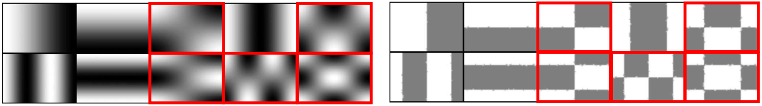
[Weiss et al. '08]

- Given a set of n data points $\mathbf{x}_i \in \mathbb{R}^d$, encode each by binary code $\mathbf{y}_i$
- 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

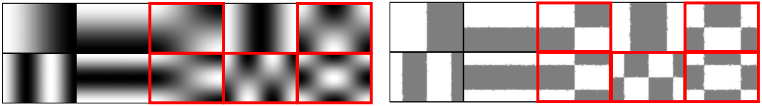
Example



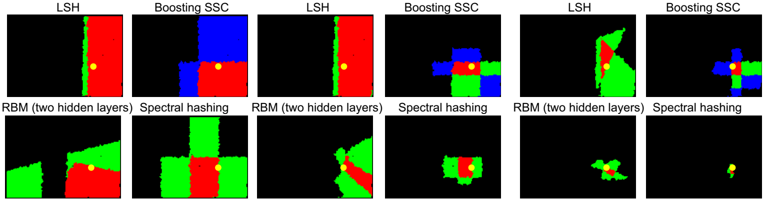
- Red: outer-product eigenfunctions: excluded
- Better to cut long dimension first
- Lower spatial frequencies are better than higher ones

Spectral hashing

Example



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



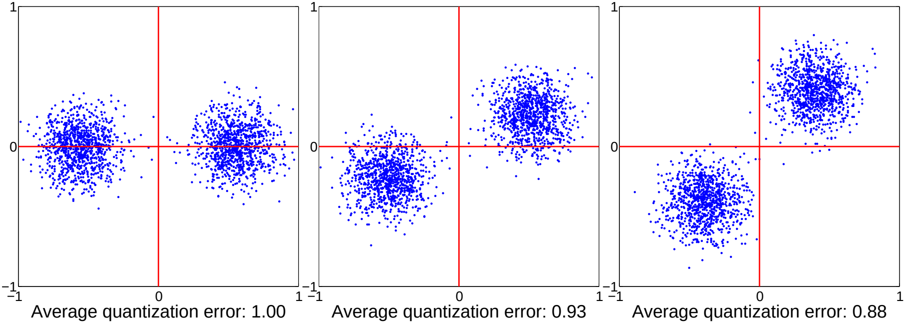
a) 3 bits b) 7 bits c) 15 bits

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

Iterative quantization

[Gong and Lazebnik '11]

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

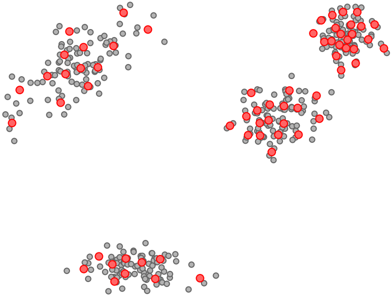


(a) PCA aligned. (b) Random Rotation. (c) Optimized Rotation.

Vector 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

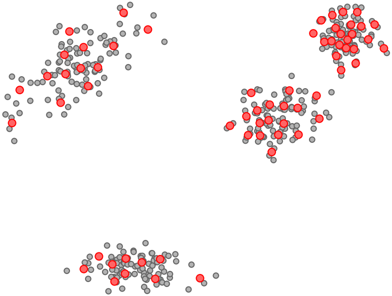
dataset

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

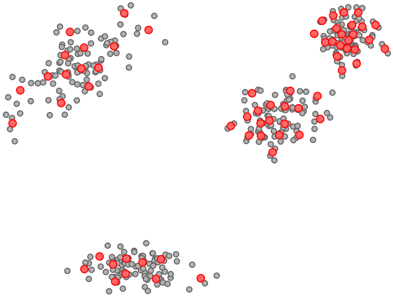
dataset

codebook

quantizer

Vector quantization

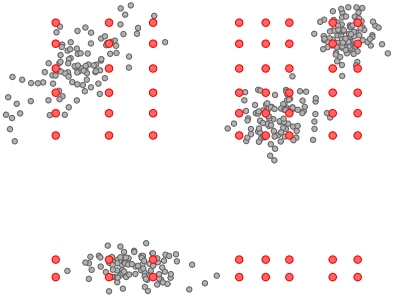
[Gray '84]



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

Product quantization

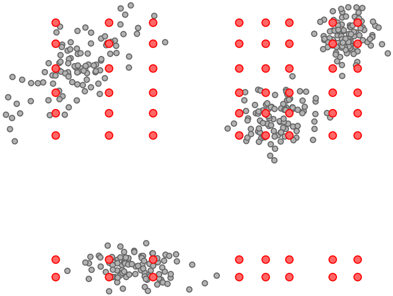
[Jégou et al. '11]



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

Product quantization

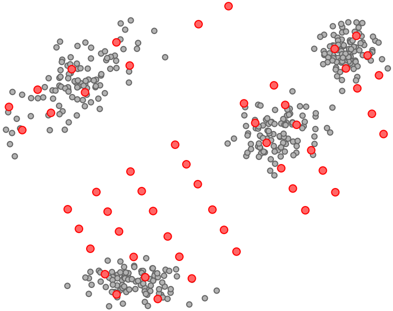
[Jégou et al. '11]



- train: $q = (q^1, \ldots, q^m)$ where $q^1, \ldots, q^m$ 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

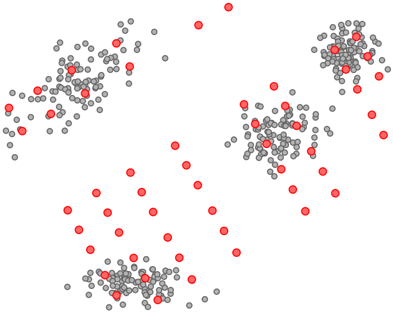
[Ge et al. '13]



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

Optimized product quantization

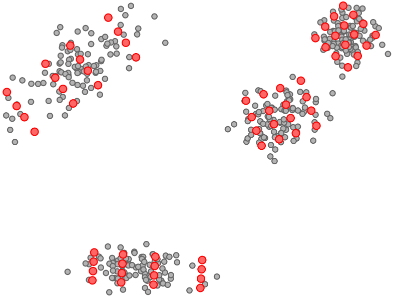
Parametric solution for $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \Sigma)$



- 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

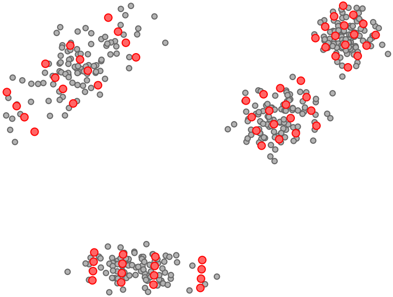
[Kalantidis & Avrithis '14]



- 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

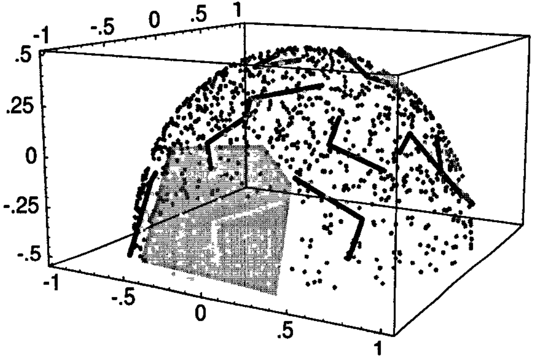
[Kalantidis & Avrithis '14]



- 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]

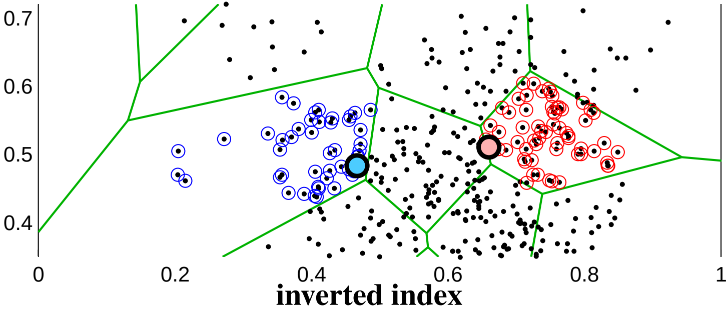


But, we are not doing dimensionality reduction!

Non-exhaustive search

Inverted multi-index

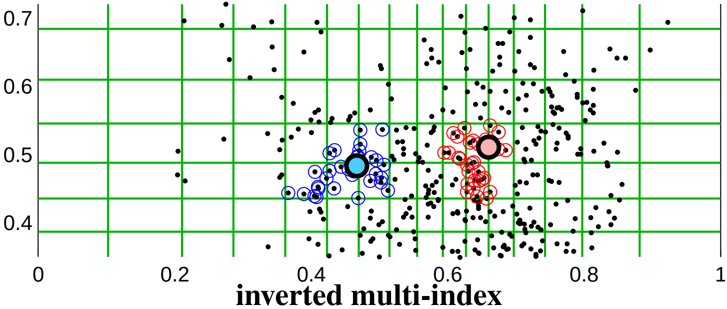
[Babenko & Lempitsky '12]



- train codebook $\mathcal{C}$ from dataset $\{\mathbf{x}_n\}$, defining a coarse quantizer Q
- quantize each point $\mathbf{x}$ to $Q(\mathbf{x})$ and encode its residual $r(\mathbf{x}) = \mathbf{x} - Q(\mathbf{x})$ by product quantizer q
- given query $\mathbf{y}$, visit w coarse cells closest to $\mathbf{y}$

Inverted multi-index

[Babenko & Lempitsky '12]

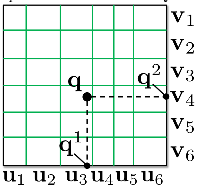


- decompose vectors as $\mathbf{x} = (\mathbf{x}^1, \mathbf{x}^2)$
- train codebooks $\mathcal{C}^1, \mathcal{C}^2$ from datasets $\{\mathbf{x}_n^1\}, \{\mathbf{x}_n^2\}$
- induced codebook $\mathcal{C}^1 \times \mathcal{C}^2$ gives a finer partition
- given query $\mathbf{y}$, visit cells $(\mathbf{c}^1,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

Inverted multi-index

Multi-sequence algorithm

space subdivision via PQ



product quantization

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

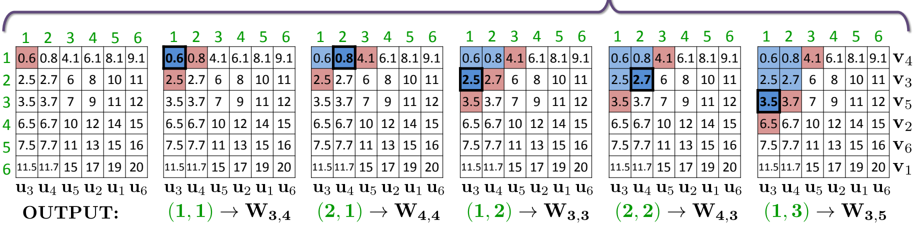
i	$\mathbf{u}_{\alpha(i)}$	r
1	$\mathbf{u_3}$	0.5
2	$\mathbf{u_4}$	0.7
3	$\mathbf{u_5}$	4
4	$\mathbf{u_2}$	6
5	$\mathbf{u_1}$	8
6	$\mathbf{u_6}$	9

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

OUTPUT:

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

$(1,1) \rightarrow \mathbf{W}_{3,4}$

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

$(2,1) \rightarrow \mathbf{W}_{4,4}$

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

$(1,2) \rightarrow \mathbf{W}_{3,3}$

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

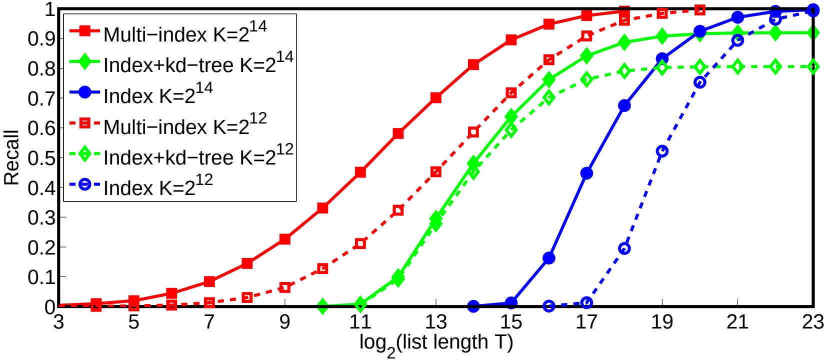
$(2,2) \rightarrow \mathbf{W}_{4,3}$

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

$(1,3) \rightarrow \mathbf{W}_{3,5}$

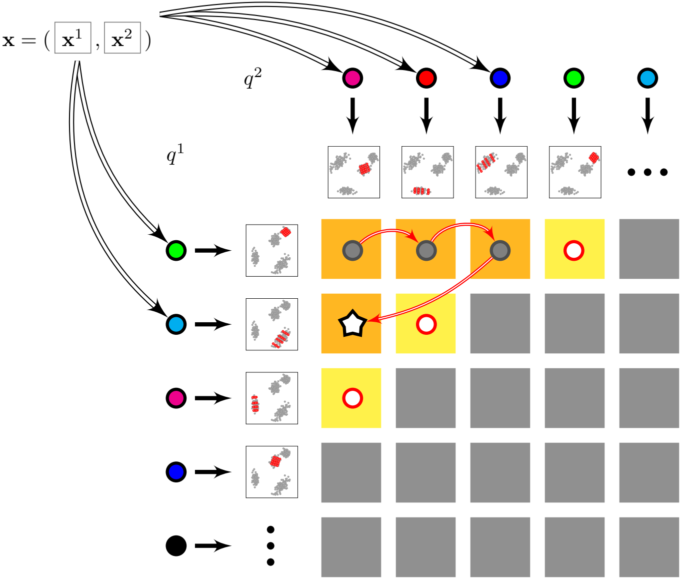
Inverted multi-index

Result on SIFT1B: are NN in candidate lists?



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

Application: image search

Deep learned image features

[Krizhevsky et al. '12] [Babenko et al. '14]

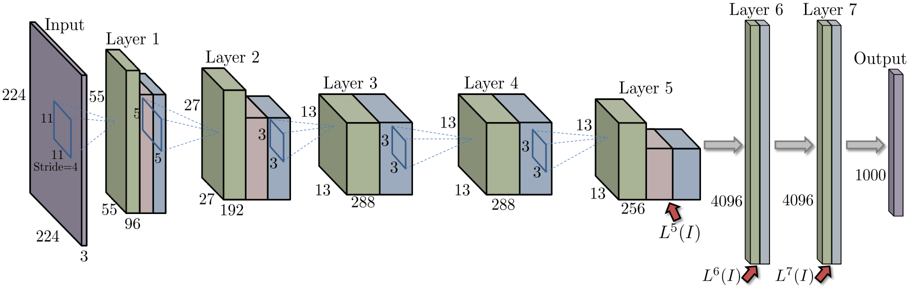
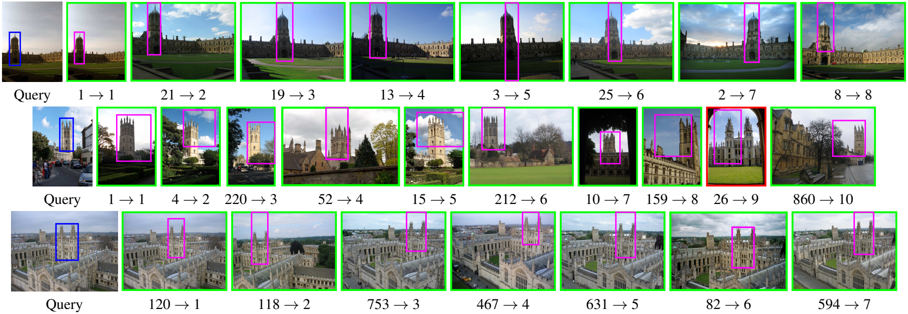


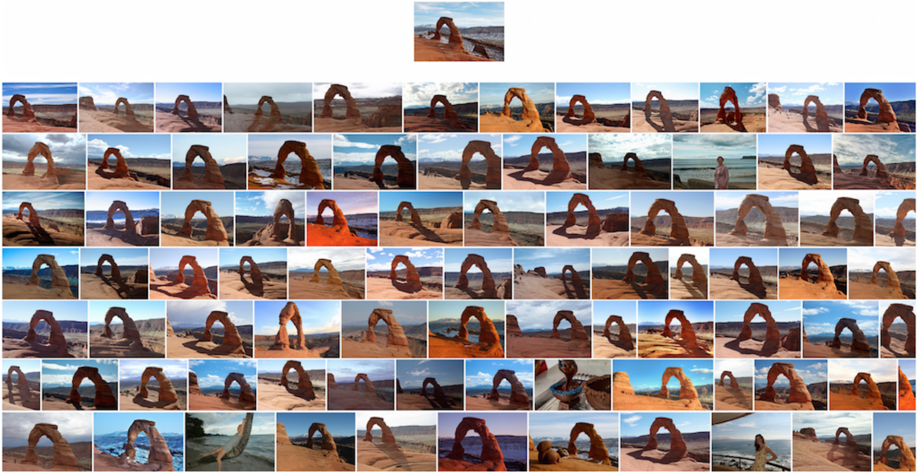
Image search on CNN activations

[Razavian '14, Babenko '15, Kalantidis '15, Tolias '16]



Multi-LOPQ on CNN activations

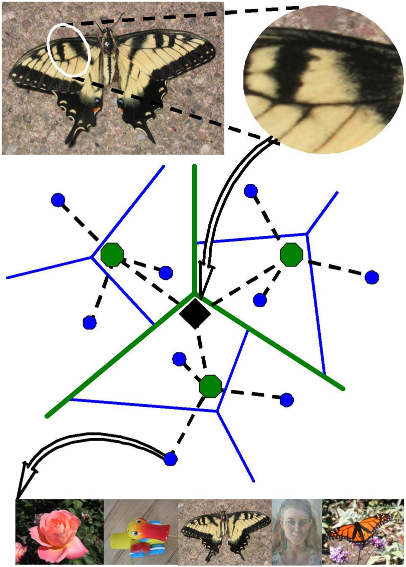
Image query on Flickr 100M (4k → 128 dimensions)



Clustering

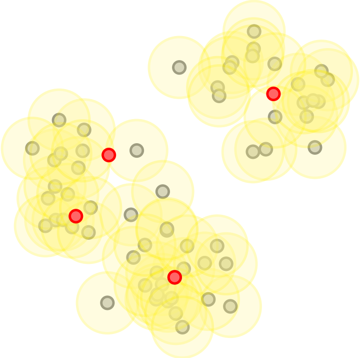
Hierarchical k-means

[Nister & Stewenius ’06]



Approximate k-means

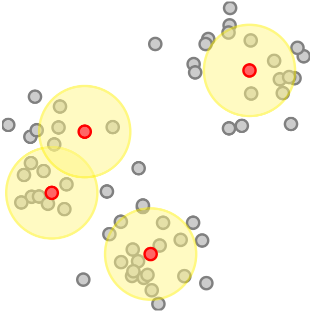
[Philbin et al. '07][Gong et al. '15]



- centroids updated as in k-means
- points assigned to centroids by approximate search
- index rebuilt in every k-means iteration

Ranked retrieval

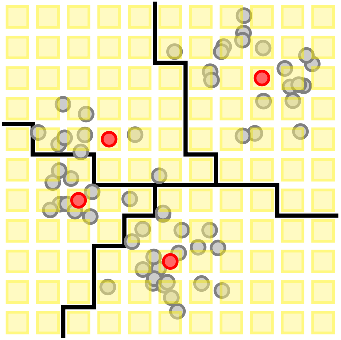
[Broder et al. '14]



- points assigned by inverse search from centroids to points
- needs conflict resolution; points may remain unassigned
- index built only once; resembles mean shift [Cheng *et al.* '95]

Dimensionality-recursive vector quantization

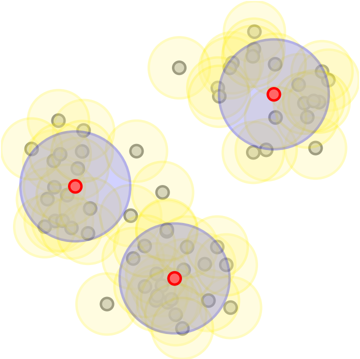
[Avrithis '13]



- points quantized as in multi-index
- cells assigned exhaustively by distance map from centroids
- points assigned by lookup

Approximate Gaussian mixtures

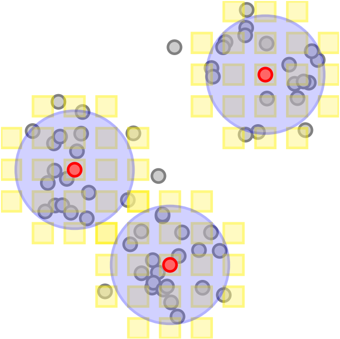
[Kalantidis & Avrithis '12]



- centroids & variances updated as in EM
- points soft-assigned by approximate search
- k dynamically estimated

Inverted-quantized k-means

[Avrithis et al. '15]



- inverse search as in RR
- points quantized as in DRVQ; search as in multi-index
- k dynamically estimated as in AGM

Inverted-quantized k-means

representation: for each cell u_α, with $X_\alpha = \{x \in X : q(x) = u_\alpha\}$

- probability $p_\alpha = |X_\alpha|/n$
- mean $\mu_\alpha = \frac{1}{|X_\alpha|} \sum_{x \in X_\alpha} x$ of all points in X_α

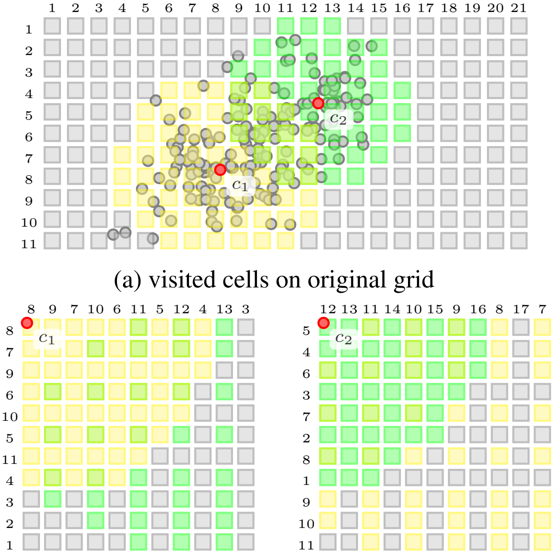
update: for each centroid c_m, with $A_m = \{\alpha \in I : a(u_\alpha) = m\}$

$$c_m \leftarrow \frac{1}{\sum_{\alpha \in A_m} p_\alpha} \sum_{\alpha \in A_m} p_\alpha \mu_\alpha,$$

assignment: for each centroid c_m,

- find the w nearest sub-codewords in each of two sub-codebooks
- run multi-sequence independently in $w \times w$ search block
- assign visited cells $m \leftarrow a(u_\alpha)$; resolve conflicts

Centroid-to-cell search

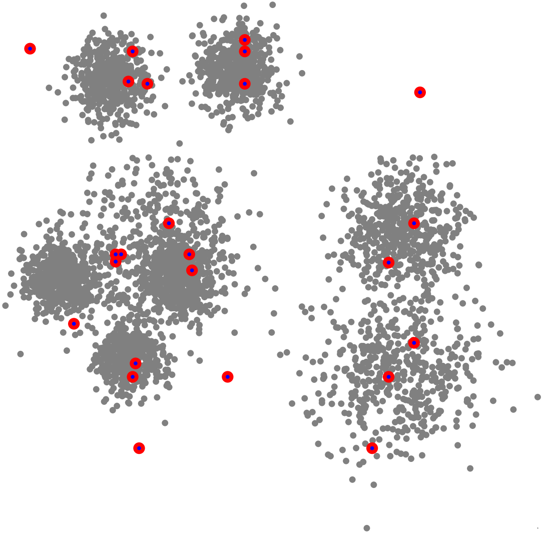


(a) visited cells on original grid

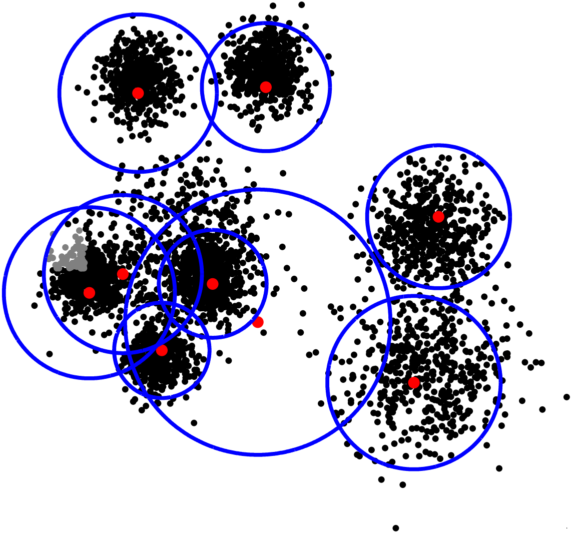
(b) search block of c_1

(c) search block of c_2

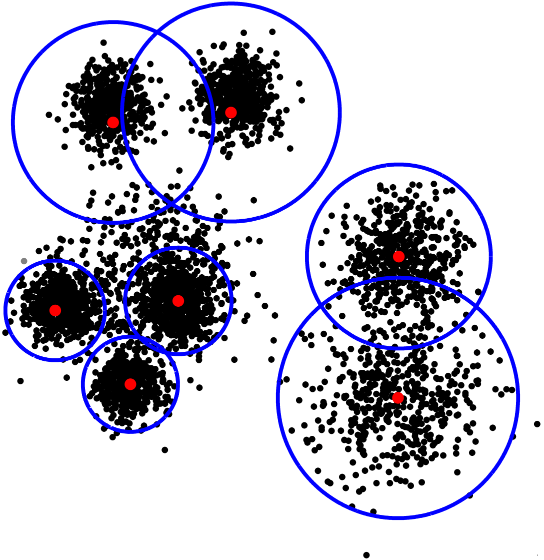
Dynamic IQ-means



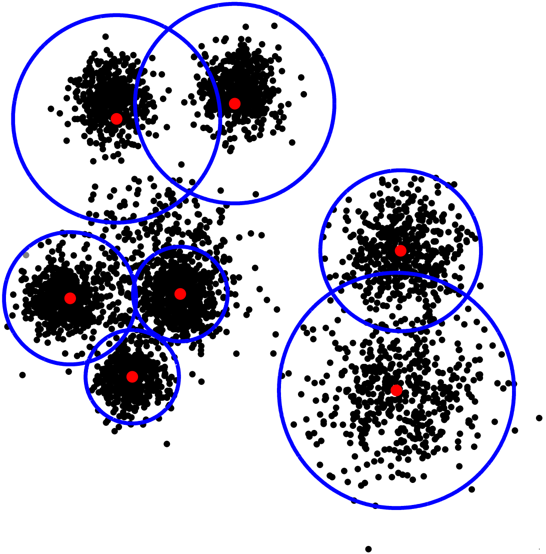
Dynamic IQ-means



Dynamic IQ-means



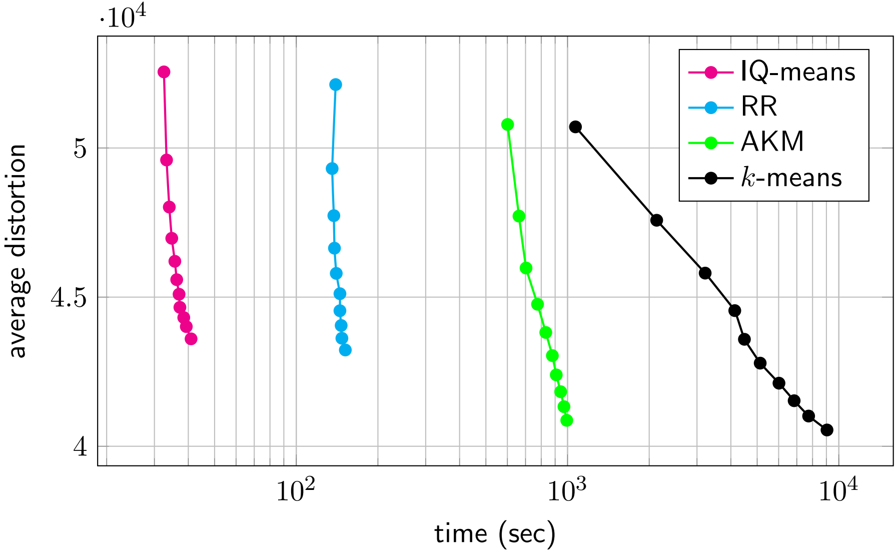
Dynamic IQ-means



Dynamic IQ-means

- quantize each centroid to closest cell just before search
- get centroid-to-centroid search at no extra cost
- greedily delete centroids as in EGM [Avrithis & Kalantidis '12]

Comparison on SIFT1M with $k \in \{10^3, \ldots, 10^4\}$



Comparison on YFCC100M, initial $k = 10^5$

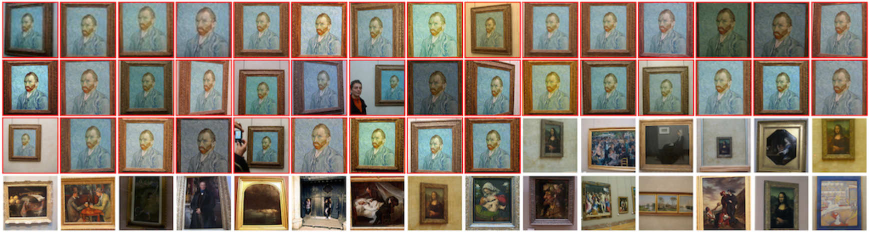
AlexNet fc7 features, 128 dimensions, optimized decomposition

	Cell-KM	DKM ($\times 300$)	D-IQ-Means
k/k'	100000	100000	85742
time (s)	13068.1	7920.0	**140.6**
precision	0.474	**0.616**	0.550

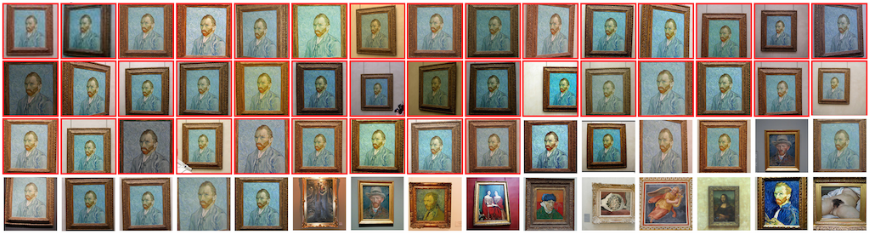
Cell-KM k-means on points quantized to cell

DKM distributed k-means on 300 machines

Mining on YFCC100M



Paris500k



Paris500k + YFCC100M

Y. Avrithis, Y. Kalantidis, E. Anagnostopoulos, I. Z. Emiris.
Web-scale image clustering revisited. ICCV 2015.

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

informatiques mathématiques

Thank you!