Web-scale image clustering revisited

Yannis Avrithis[†], Yannis Kalantidis[‡], Evangelos Anagnostopoulos[†], Ioannis Z. Emiris[†]

[†]University of Athens, [‡]Yahoo Labs San Francisco

Related ideas & challenges

Problem formulation

- given a dataset X of n points in $\mathbb{R}^d$, find k cluster centroids minimizing distortion
- k-means
  - **assignment step:** for every point, find closest centroid
  - **update step:** given point assignments, update centroids

Approximations & speed-ups

- the assignment step is the bottleneck
- **approximate** k**-means** [Philbin *et al.* CVPR, 2007]: ANN to speed-up assignment step
- **binary** k**-means** [Gong *et al.* CVPR, 2015]: binarize points and centroids

Inverse search

- data remain fixed across iterations: index points, search for centroids
- search independently **ranked retrieval** [Broder *et al.* WSDM, 2014]

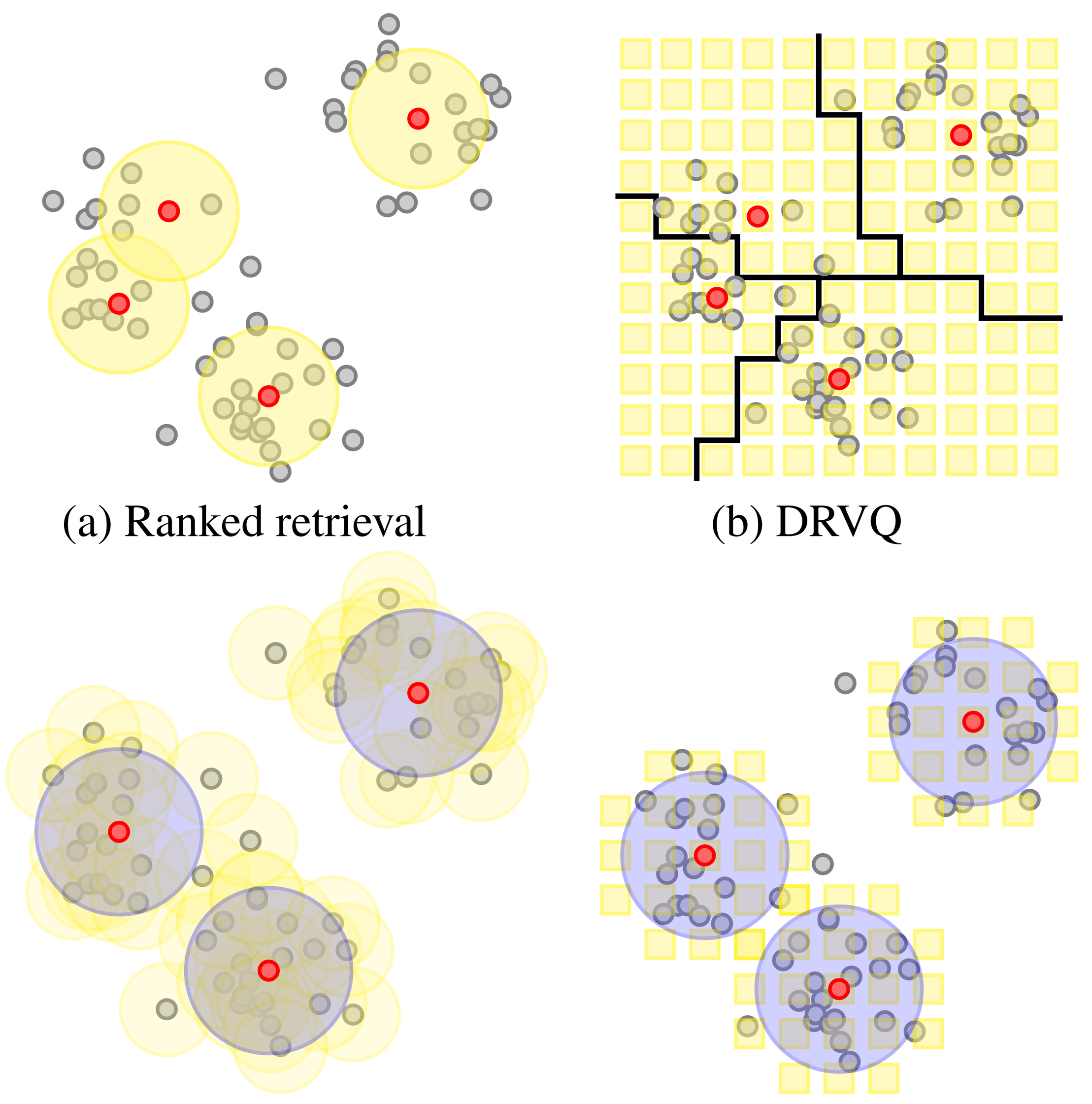
Compression of points and/or centroids

- **dimensionality recursive vector quantization (DRVQ)** [Avrithis, ICCV, 2013]
  - borrows ideas from *inverted multi-index* [Babenko & Lempitsky, 2012]
  - quantize points to centroids, adopt inverse search
  - search is a propagation on a 2d-grid, each cell visited only once, joint priority queue
- **binary** k**-means** [Gong *et al.* CVPR, 2015]: binarization of points & centroids

Dynamic estimation of k

- **expanding Gaussian mixtures (EGM)** [Avrithis & Kalantidis, ECCV, 2012]
  - estimation of cluster overlap, point-to-centroid search

Inverted Quantized k-means (IQ-means)



(a) Ranked retrieval (b) DRVQ

(c) EGM (d) This work: IQ-means

- subspace quantization & search via multi-index
- inverse search, independent queries per centroid
- dynamic estimation of k

Quantization & Representation

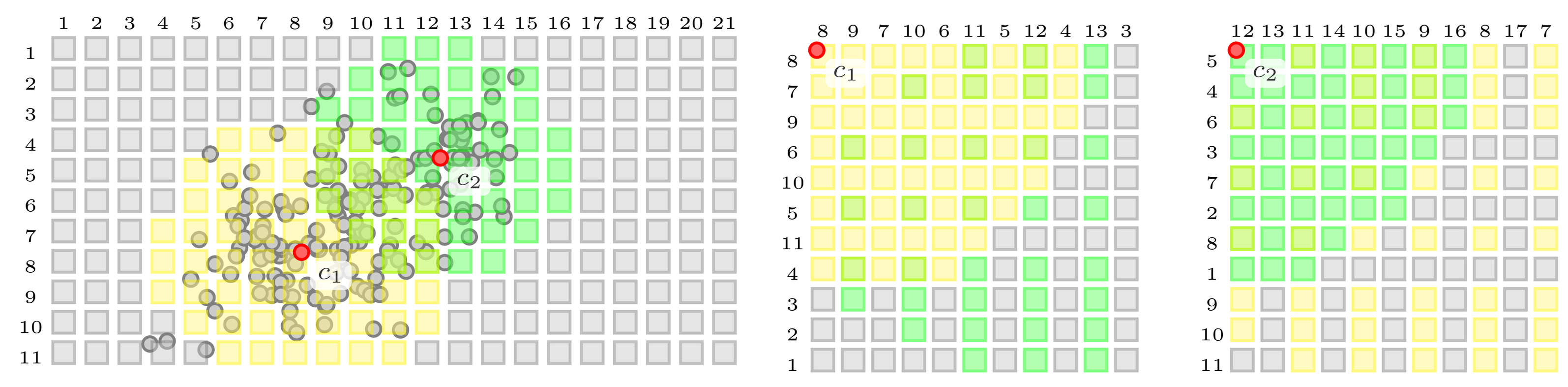
Compressing the dataset

- express $\mathbb{R}^d$ as the Cartesian product of two orthogonal subspaces, $S^1 \times S^2$, of $d/2$ dimensions each – subject to optimization [Ge *et al.* , 2013]
- train two sub-codebooks U^1, U^2 of size s independently on projections of sample data on S^1, S^2
- codebook $U = U^1 \times U^2$ contains $s \times s$ *cells* – can be seen as a discrete two dimensional *grid* [Babenko & Lempitsky, 2012]
- vector $x = (x^1, x^2)$ can be quantized to a cell $q(x) = (q^1(x^1), q^2(x^2))$, where $q^\ell(x^\ell) = \arg\min_{u^\ell \in U^\ell} \|x^\ell - u^\ell\|$ for $\ell = 1, 2$

Discarding original data

- for cell u_α, probability $p_\alpha = |X_\alpha|/n$, with $X_\alpha = \{x \in X : q(x) = u_\alpha\}$
- the mean $\mu_\alpha = \frac{1}{|X_\alpha|}\sum_{x \in X_\alpha} x$ of all points in X_α is kept for each cell u_α
- cells with their sample mean μ_α and probability p_α replace the original data
- an arbitrary initial set C of k centroids is assumed

Centroid to cell search



visited cells on original grid search blocks for c_1, c_2

Update Step

Moving the centroids

- for all $c_m \in C$:

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

where:

- $A_m = \{\alpha \in I : a(u_\alpha) = m\}$: the indices of all cells assigned to c_m
- $P_m = \sum_{\alpha \in A_m} p_\alpha$: the proportion of points assigned to centroid c_m, with $a(u) = \arg\min_{c_m \in C} \|u - c_m\|$

Assignment step

Multi-index search independently for every centroid

- **centroid-to-cell search**: index cell means, search independently for every centroid
- for each centroid c_i, the w nearest sub-codewords are found in U^1, U^2, and ordered by ascending distance to c_i, for $i = 1, 2$
- a $w \times w$ *search block* is thus determined for c_i
- the **multi-sequence** [Babenko & Lempitsky, 2012] algorithm is used for traversing the cells in the search block
- **termination:** count the total number of underlying points in visited cells, and terminates when this reaches a target number T

Dynamic estimation of k

Centroid-to-centroid search

- centroids can still be quantized on the grid, just like data points
- quantize each centroid c_m to cell u_α, record its index m in *cen*$[\alpha]$
- at no extra cost you can have a list of neighboring centroid indices and distances for each centroid m

Centroid modeling

- following EGM [Avrithis & Kalantidis, ECCV, 2012], we model the distribution of points assigned to cluster c_m by an isotropic normal density $\mathcal{N}(x|c_m, \sigma_m)$, where

$$\sigma_m^2 \leftarrow \frac{1}{P_m} \sum_{\alpha \in A_m} p_\alpha \|\mu_\alpha - c_m\|^2.$$

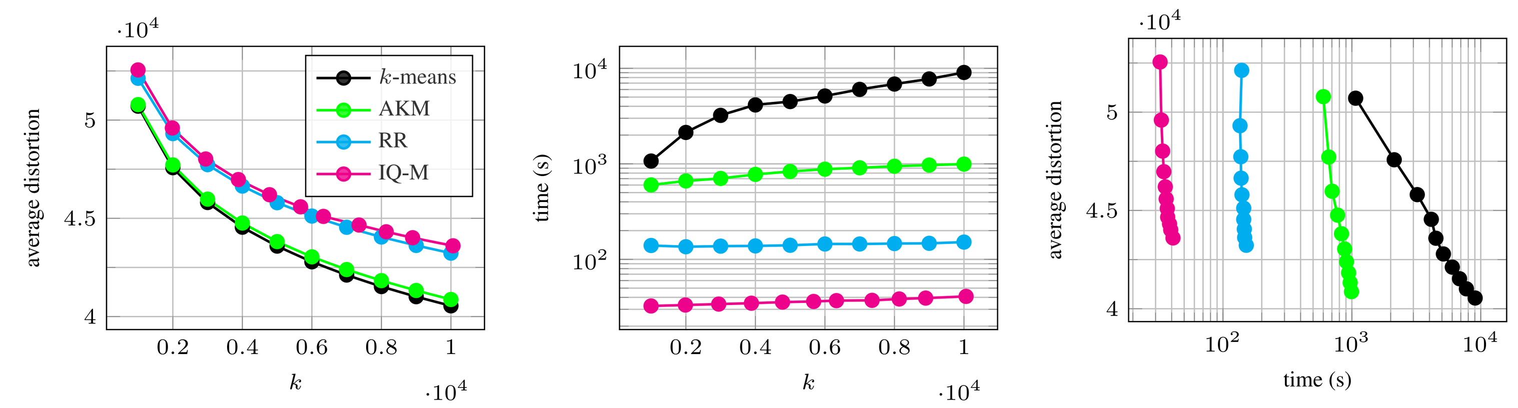
Centroid purging

- iterate over all clusters m in descending order of population P_m
- for every centroid, get list of neighboring centroids and compute overlap
- purge clusters that overlap too much with all clusters kept so far

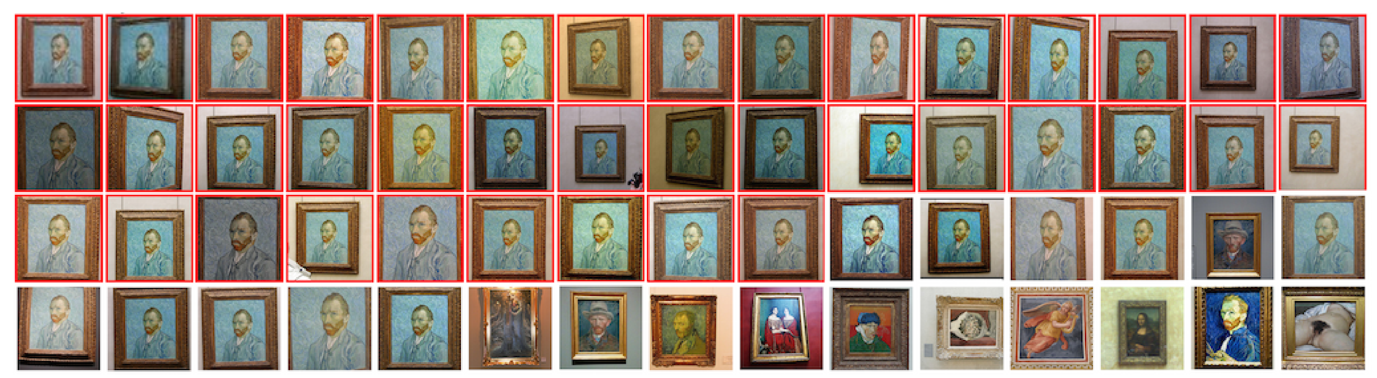
Experiments

- **Datasets:** SIFT1M, Paris, Yahoo Flickr Creative Commons 100M (YFCC100M)
- **Image representation**: AlexNet CNN fc7 features, PCA to 128 dimensions, optimized subspace decomposition [Ge *et al.* , 2013]
- **Metrics:** distortion, timings and cluster precision (or *purity*) on a noisy set of image classification labels

SIFT1M



Paris



YFCC100M

	CKM	Distributed$_{\times 300}$	D-IQ-means
k/k'	100000	100000	85742
time (s)	13068.1	7920.0	**140.6**
precision	0.474	**0.616**	0.550

Table: time per iteration and average precision, initial $k = 10^5$, s=8192

Cluster 100M images in less than an hour on a single machine

Source code: `http://image.ntua.gr/iva/research/lopq/`

Contact: iavr@image.ntua.gr, ykal@yahoo-inc.com, aneva@math.uoa.gr, emiris@di.uoa.gr