Recent Image Search Techniques

CVPR 2016 Tutorial

Sung-Eui Yoon and **Zhe Lin**

Outline

- Indexing and Encoding Schemes for Large-Scale Image Search (45 min)
  - Product quantization and its variants
  - Inverted index, inverted multi-index, residual-shortlist
- Applications of Image Search (45 min)
  - Object retrieval and localization
  - Facial attribute recognition
  - Discriminative feature learning with CNN
  - Large-scale semantic search, recommendation
  - Large-scale image tagging

Indexing/Encoding Schemes

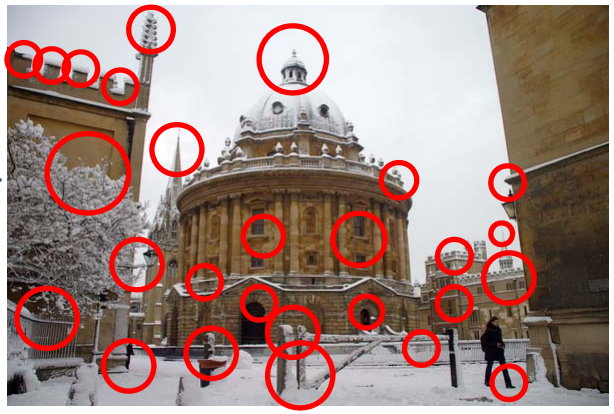
- Local descriptor-based method
  - Keypoints + Local invariant descriptors [Lowe99, Mikolajczyk04]
  - Bag of features model [Sivic and Zisserman03] [Philbin07]
  - Inverted index [Sivic and Zisserman03] [Nister06]
  - Geometric verification [Philbin07] [Jegou08]

- Global descriptor-based method
  - Product quantization [Jegou11]
  - Optimized product quantization [Ge13]
  - Distance-encoded product quantization [Heo14]
  - Inverted Multi-Index [Lempitsky13]
  - Shortlist computation [Heo16]

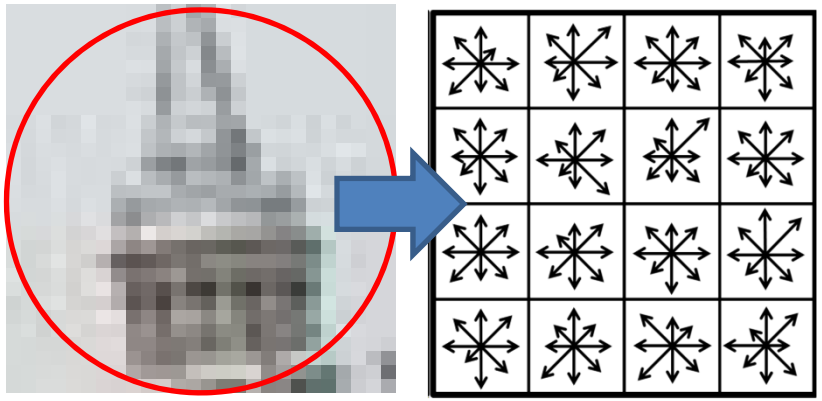
Local Feature-based Mathod

Feat. Extr.

Interest points



Local descriptors



Quantize

Bag of features

Photo source: Lempitsky's slides

Local Feature-based Method

- Mainly used for identical object/scene insta nce retrieval

Images from H. Jegou's SSMS'12 Talk Slide

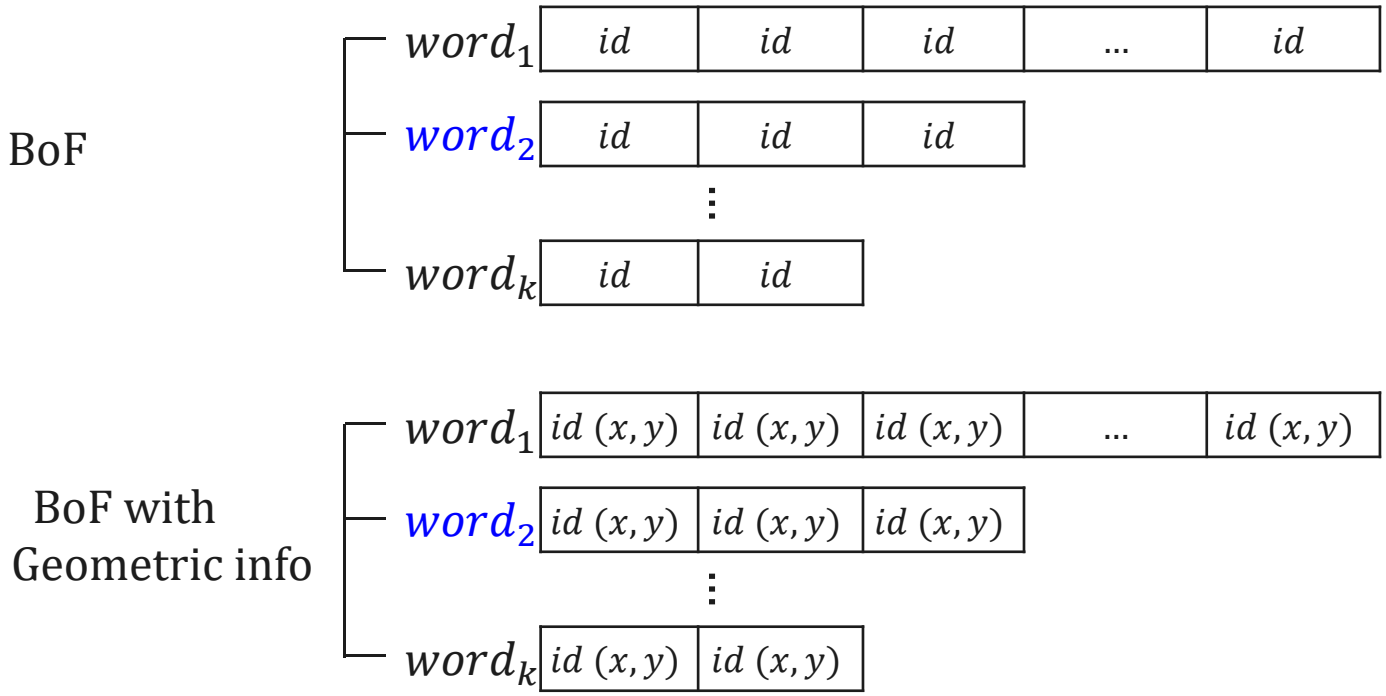
[X. Shen et al. CVPR 2012]

Local Descriptor-based Method

- How to index bag-of-features ?
  - Build a codebook by k-means
  - Encode each local descriptor into a visual word
  - Store visual words into an **inverted file**

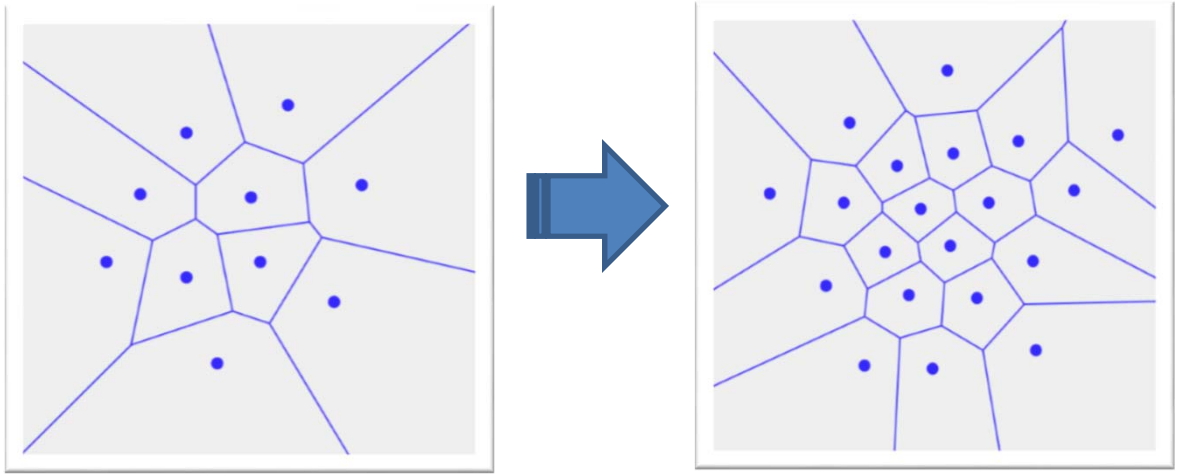
Inverted Index

- Organize bag-of-features w.r.t. visual words



Vocabulary Size

- Larger codebooks for efficiency ?
  - Lower quantization errors with increasing the voc. size
  - Increased assignment complexity and memory requirement
  - Tradeoff between assignment accuracy and efficiency



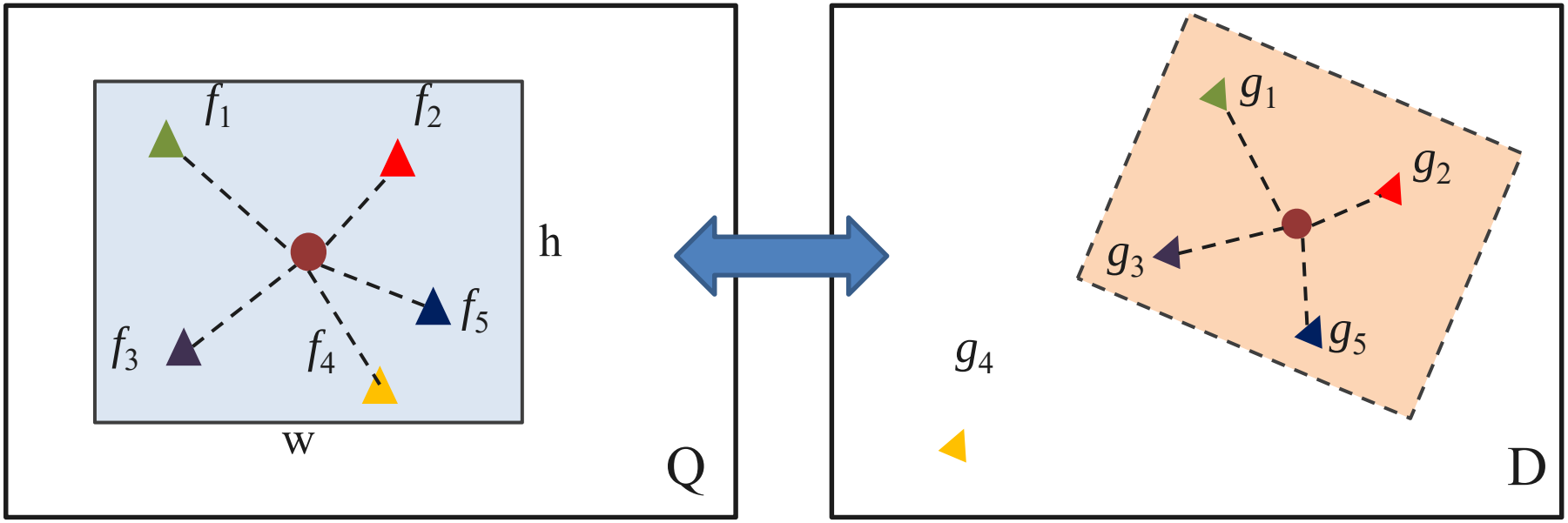
Learning Large Vocabularies

- Hierarchical k-means [Nister06]
  - Quantize recursively

- Approximate k-means [Philbin07]
  - Flat k-means with fast approximate assignment step with randomized tree search

Geometric Verification

- RANSAC-based method [Philbin07]
- Weak geometric consistency [Jegou08]
- Geometrical min-hash [Chum 09]
- Bundling features [Wu 09]
- Spatial inverted file / Local BoW [Lin 10]
- Geometry preserving visual phrases [Zhang 11]
- Generalized Hough-voting-based [Shen 12]



Scalability

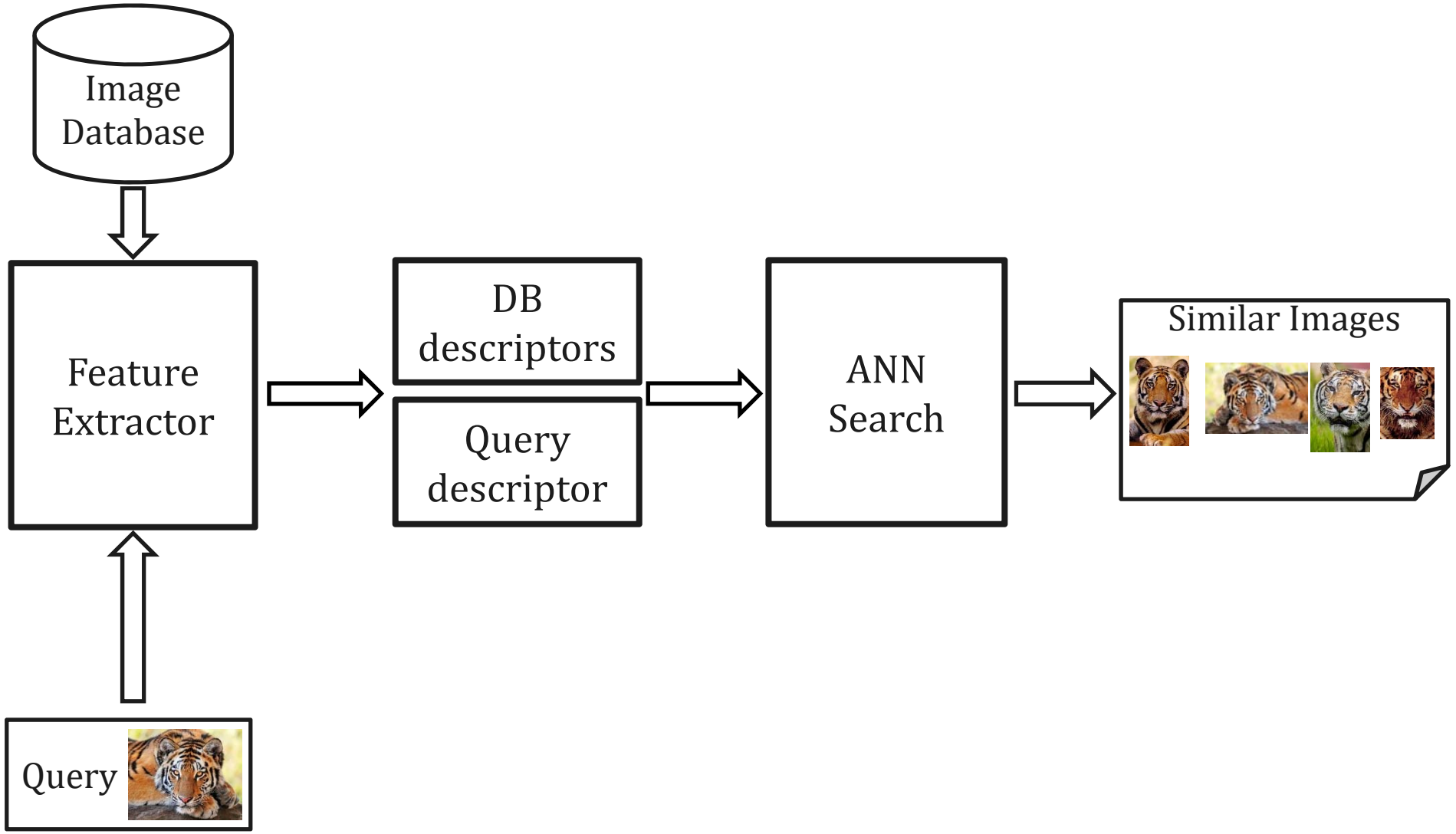
- Limited to a few million images due to mem ory cost

- How about larger scale search ?
  - Search on a database of 10M to 1B
    - Global descriptor
    - Indexing (coarse quantizer) -- speed
    - Encoding (compact representation) -- memory

What is important for very large-scale search ?

- Performance criteria
  - Search accuracy
  - Search speed
  - Memory/storage usage

Global Feature-based Method



Nearest Neighbor Search

- Exhaustive NN Search
  - Euclidean distance metric

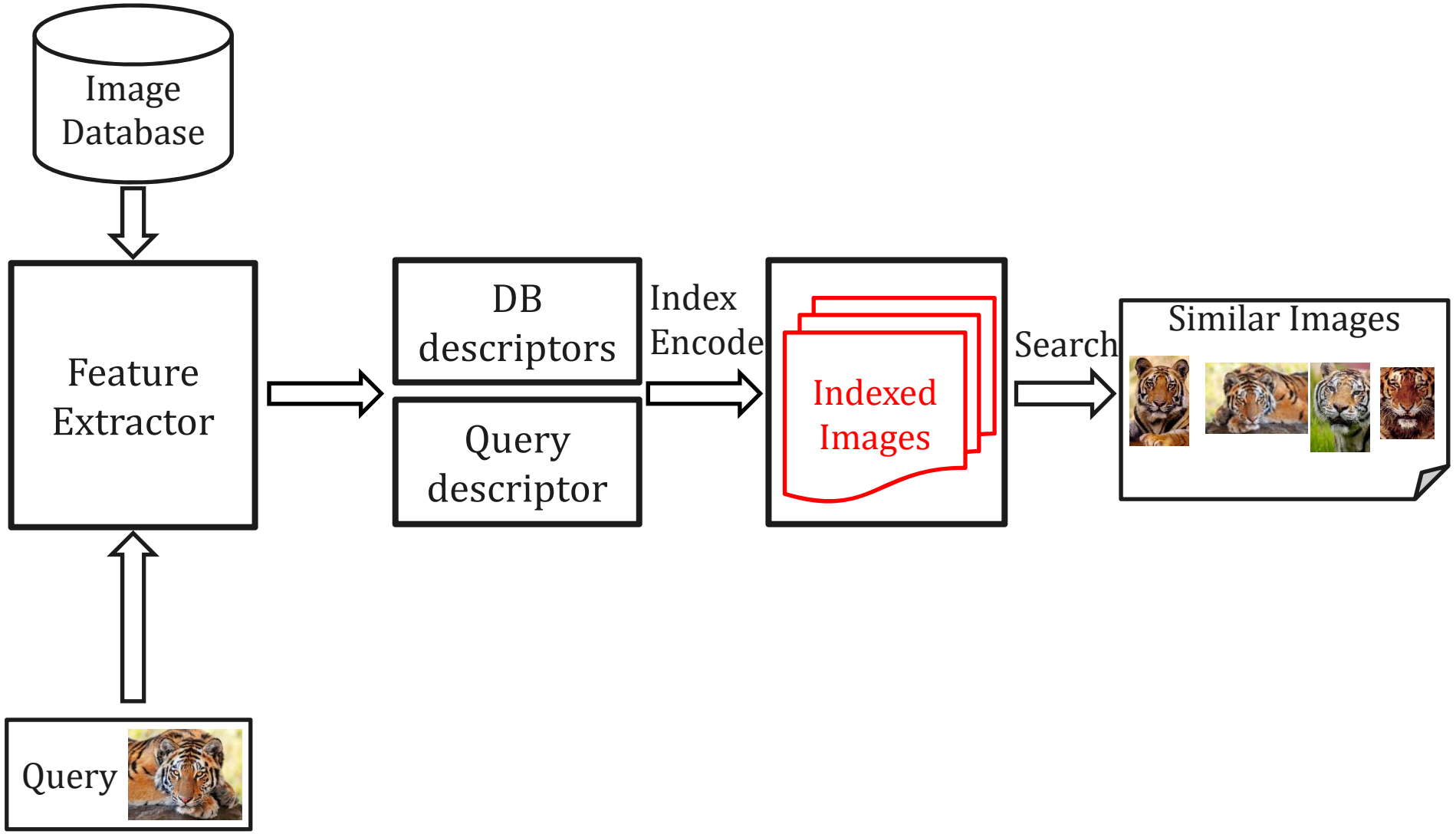
$$\mathrm{NN}(x) = \arg\min_{y \in \mathcal{Y}} ||x - y||^2$$

  - Very costly when the DB is very large, and the d escriptor is very high dimensional: O(n*d)
  - Can be used as ”verification” method on a small candidate set

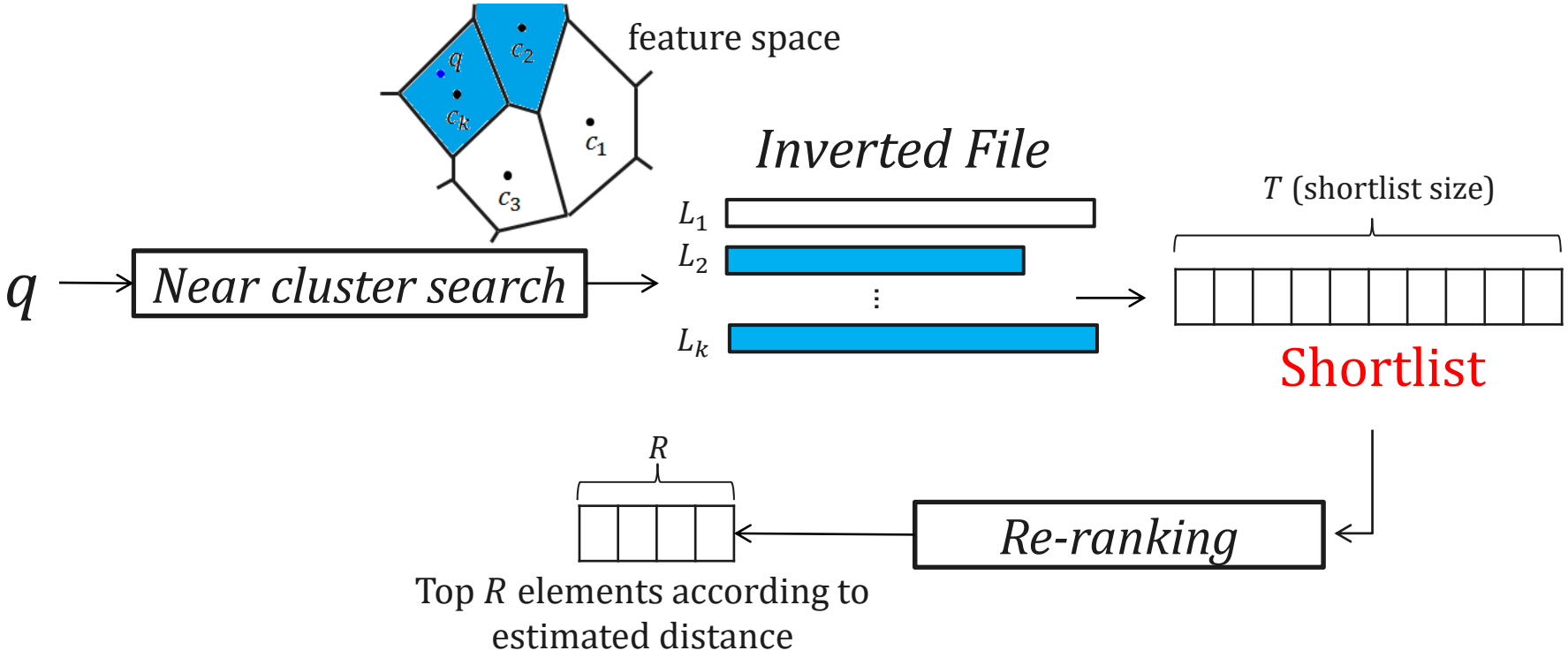
Efficient Solutions

- Dimensionality reduction
  - e.g. PCA
- Hierarchical methods (ex: kd-tree, HKM)
  - Efficient and accurate for a mid-range dataset
  - Suffer from 'curse of dimensionality'
  - Do not provide a compact data representation
  - e.g. FLANN library [Muja & Lowe 09]
- Binary code embedding
  - LSH, spectral hashing, min-hash, hamming embedding, etc.
  - Very fast during search time
  - But, may not handle well very high-dim data
  - Requires raw features in verification, e.g. 128GB for 1B SIFT
- Quantization-based methods (*)
  - **Indexing:** inverted indexing, inverted multi-indexing, etc.
  - **Encoding:** vector quantization, product quantization, optimized product quantization, distance-encoded product quantization, etc.

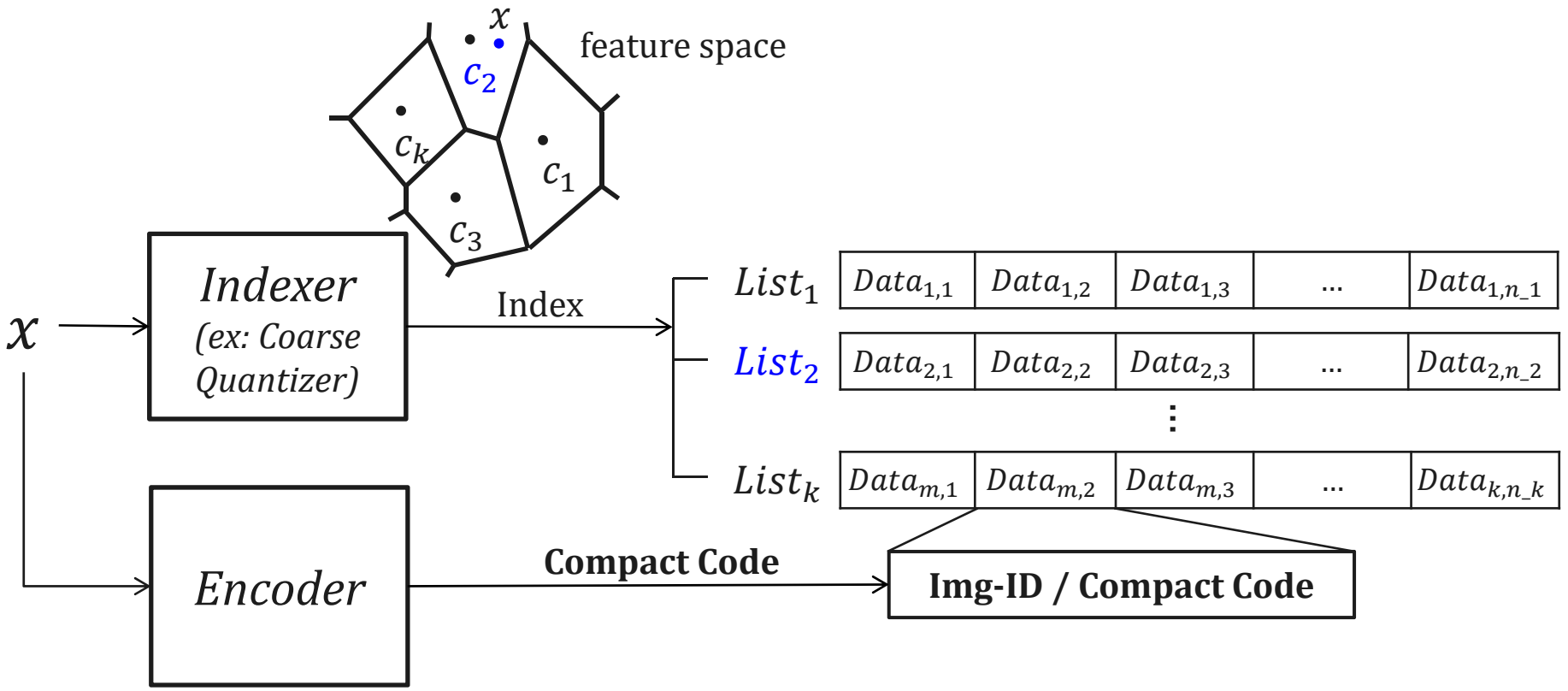
Global Descriptor-based Method



Search Framework (Global desc.)



Indexing and Encoding



Indexing and Encoding

- **Encoding**
  - Residual vector as input to encoder
  - Product quantization and its variants

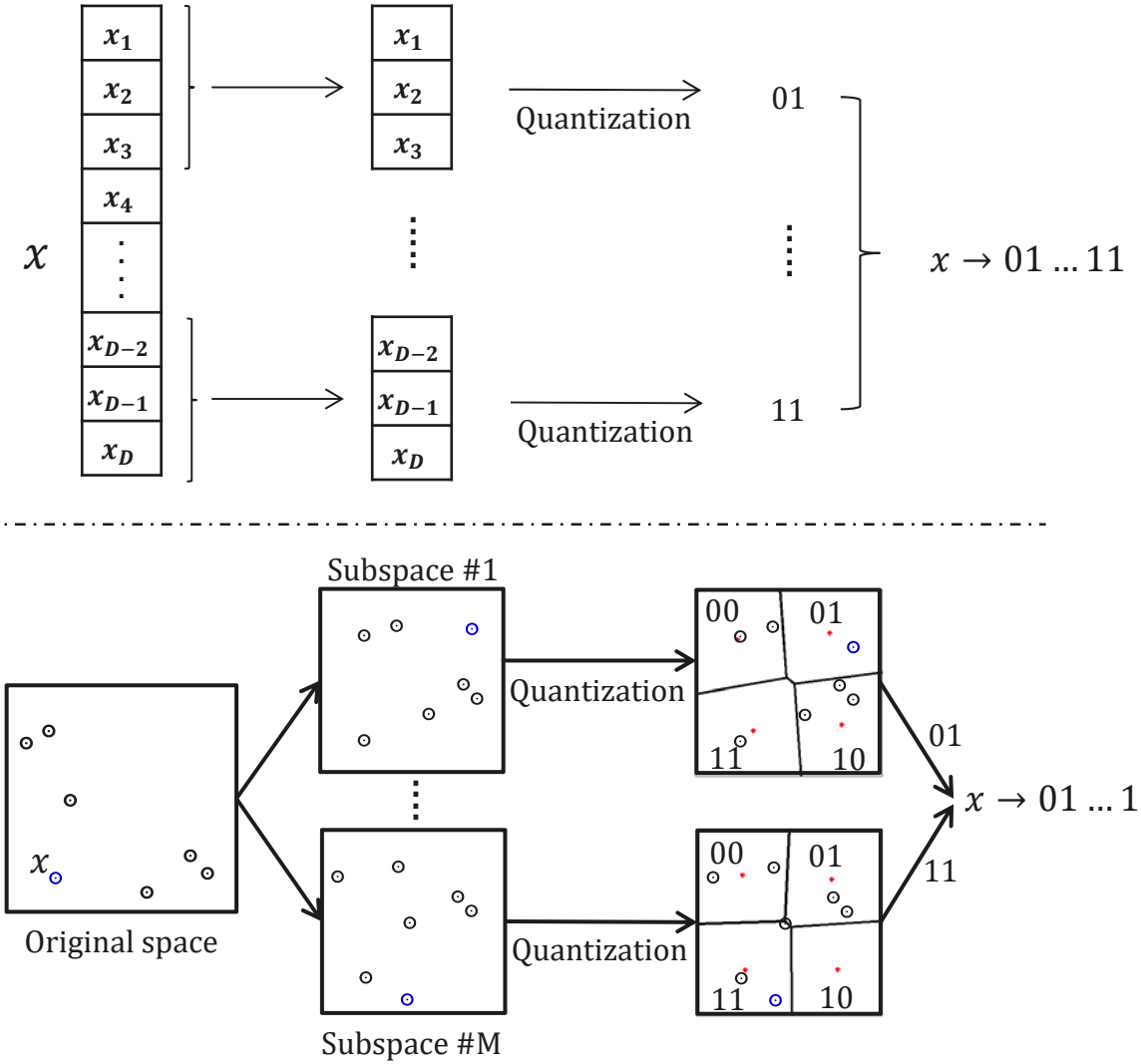
- **Indexing**
  - Inverted (multi-)index
  - Residual-aware shortlist selection

Product Quantization [Jegou et al., TPAMI 2011]

- Vector quantization
  - For a very large codebook, e.g. K = 4^64
    →intractable as an encoder (speed and memory)

- Product quantization
  - Cartesian product of subspace quantization
  - Can generate an exponentially large codebook at very low memory/time cost

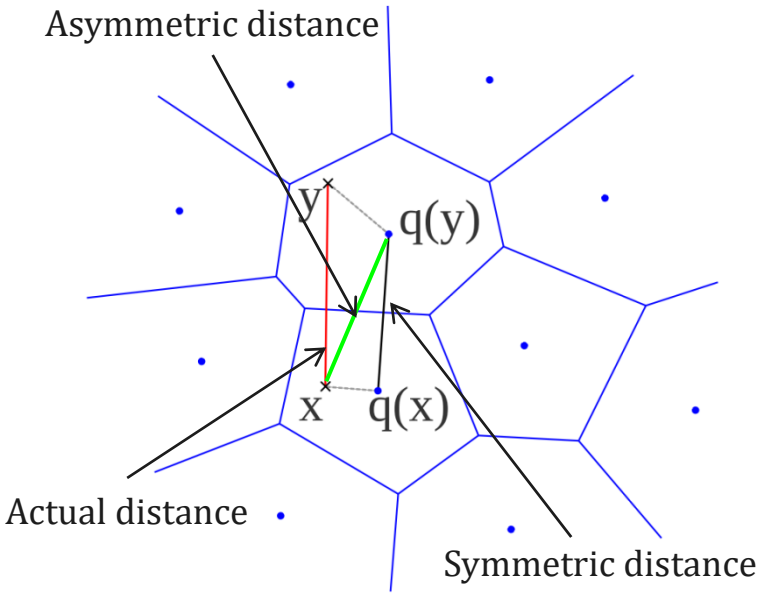
Product Quantization [Jegou et al., TPAMI 2011]



Product Quantization [Jegou et al., TPAMI 2011]

Distance estimation

(between encoded y and query x)



x: query, y: data

$q(x)=c_x$

$q(y)=c_y$

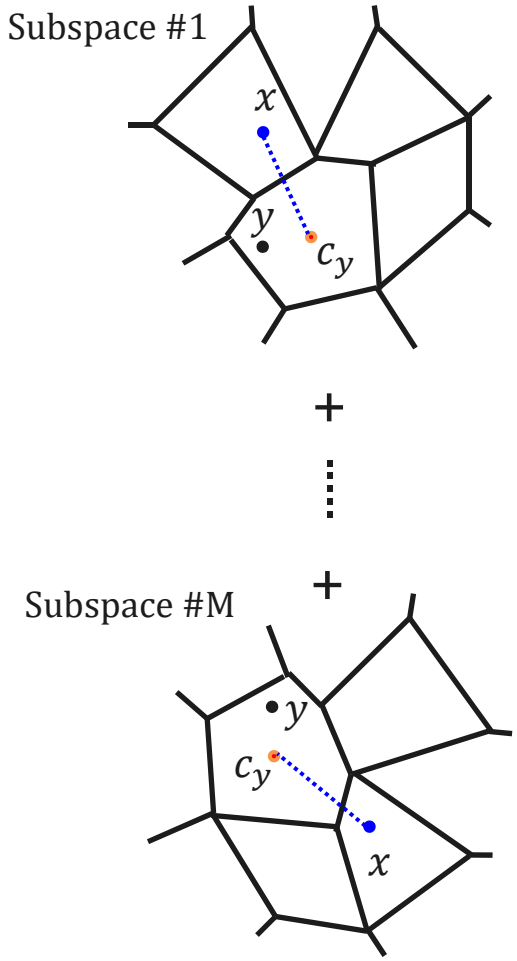
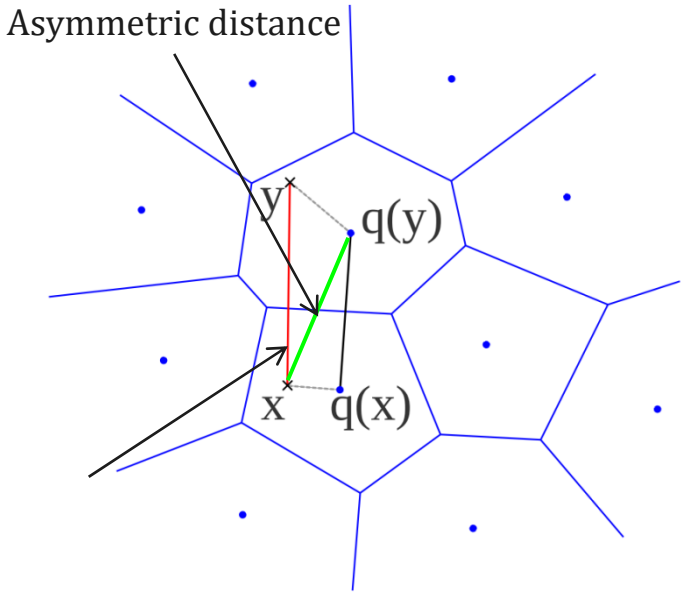


Figure from [Jegou et al., TPAMI 2011]

Distance Estimation Error

- Distance estimation error is statistically bounded by quantization error [Jegou PAMI 2011]
  - MSDE(q) <= MSE(q)

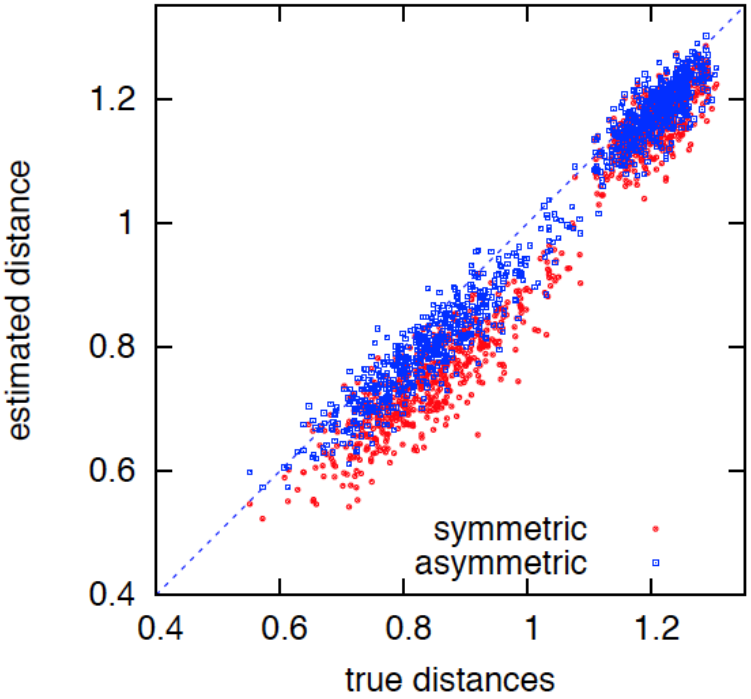
$$\Big(d(x,y) - d(x,q(y))\Big)^2 \leq d\big(y,q(y)\big)^2$$

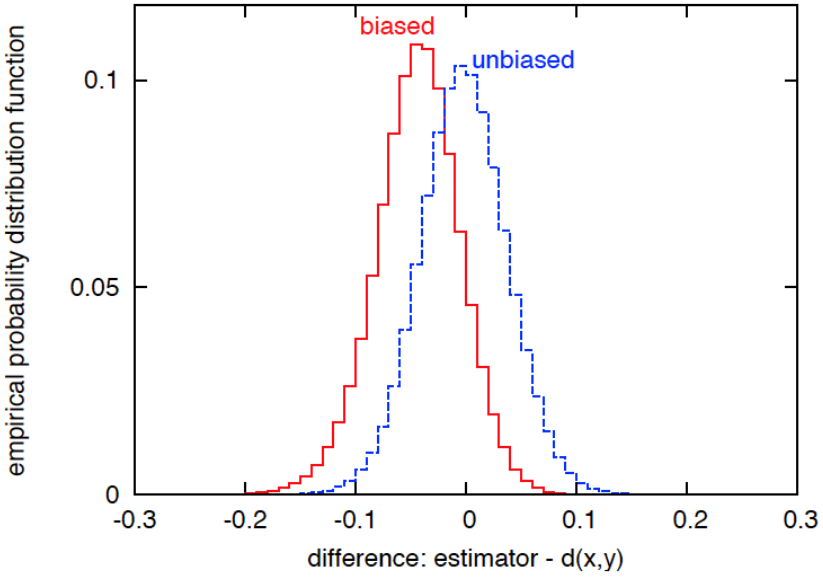


Distance Estimation Bias

- Unbiased asymmetric estimator

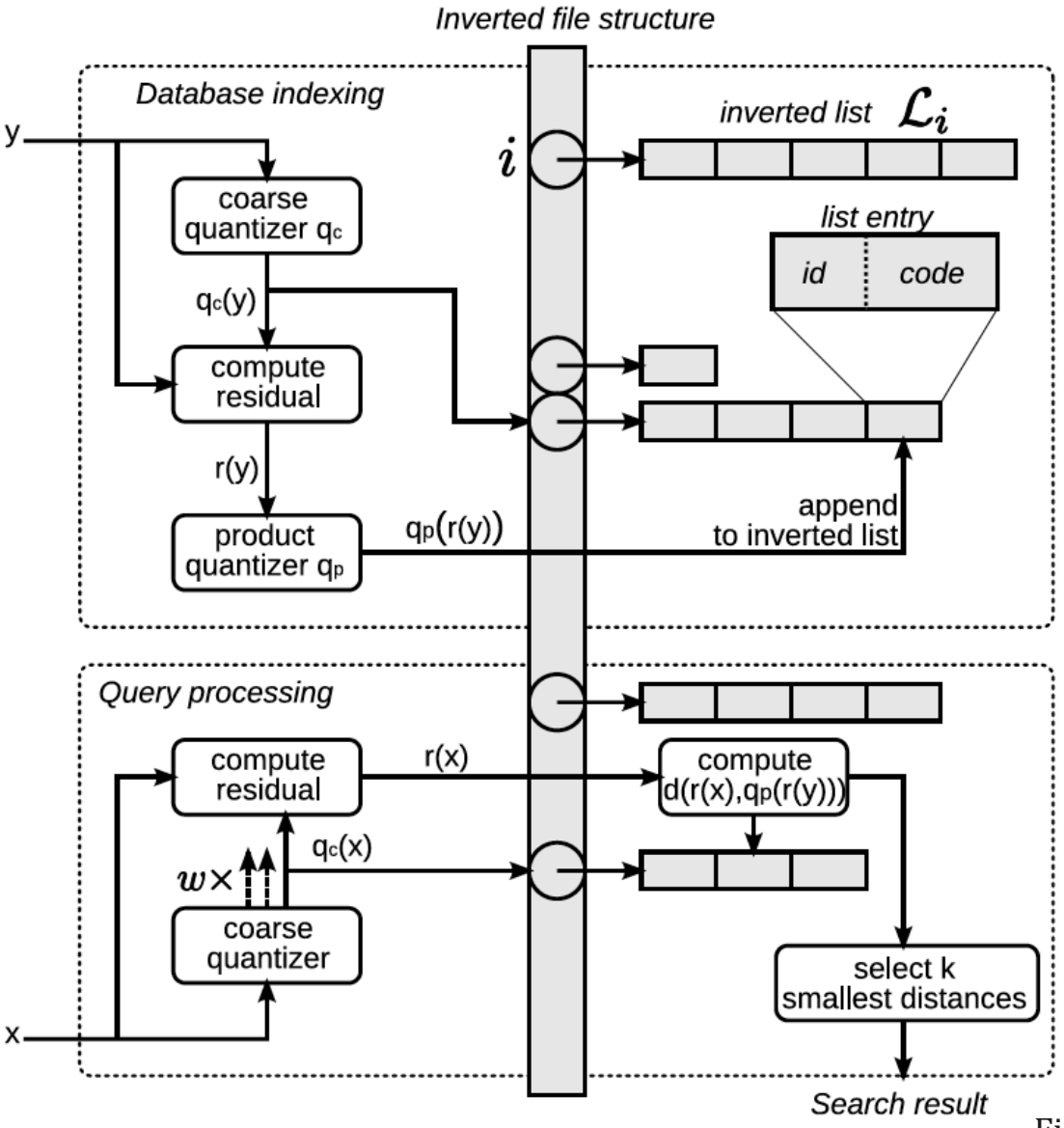
$$\tilde{e}(x,y) = \tilde{d}(x,y)^2 + \sum_j \xi_j(y)$$





Figures from [Jegou PAMI 2011]

Non-Exaustive Search (IVFADC)

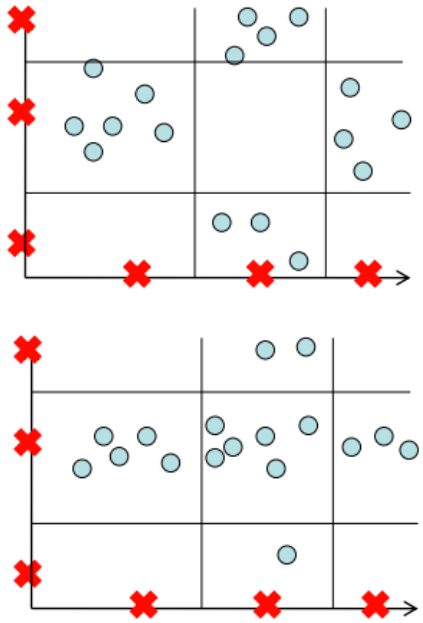


Figures from [Jegou PAMI 2011]

Optimized Product Quantization

[Ge et al., CVPR 2013]
[Norozi and Fleet, CVPR 2013]

- Problem of product quantization
  - Subspaces are assumed to be independent
  - Subspaces have unbalanced variances
- Subspace decomposition



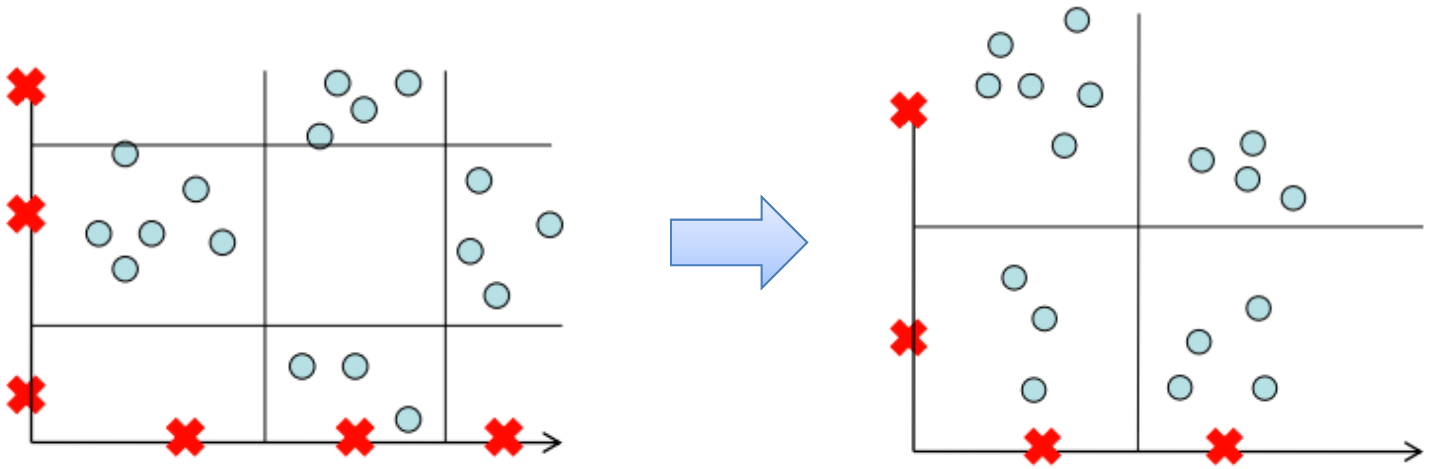
Figures from Arandjelovic's slides

	SIFT		GIST
m	4	8	8
natural	0.593	0.921	0.338
random	0.501	0.859	0.286
structured	0.640	0.905	0.652

Optimized Product Quantization

[Ge et al., CVPR 2013]
[Norozi and Fleet, CVPR 2013]

- Optimal subspace decomposition
  - Estimate a rotation projection matrix R to minimize quantization distortion
  - Rotation can de-correlate data and balance subspac e variances well



From Arandjelovic's slides

Optimized Product Quantization

[Ge et al., CVPR 2013]
[Norozi and Fleet, CVPR 2013]

- Formulation

$$\min_{R,\mathcal{C}^1,\ldots,\mathcal{C}^M} \sum_{\mathbf{x}} \|\mathbf{x} - \mathbf{c}(i(\mathbf{x}))\|^2,$$

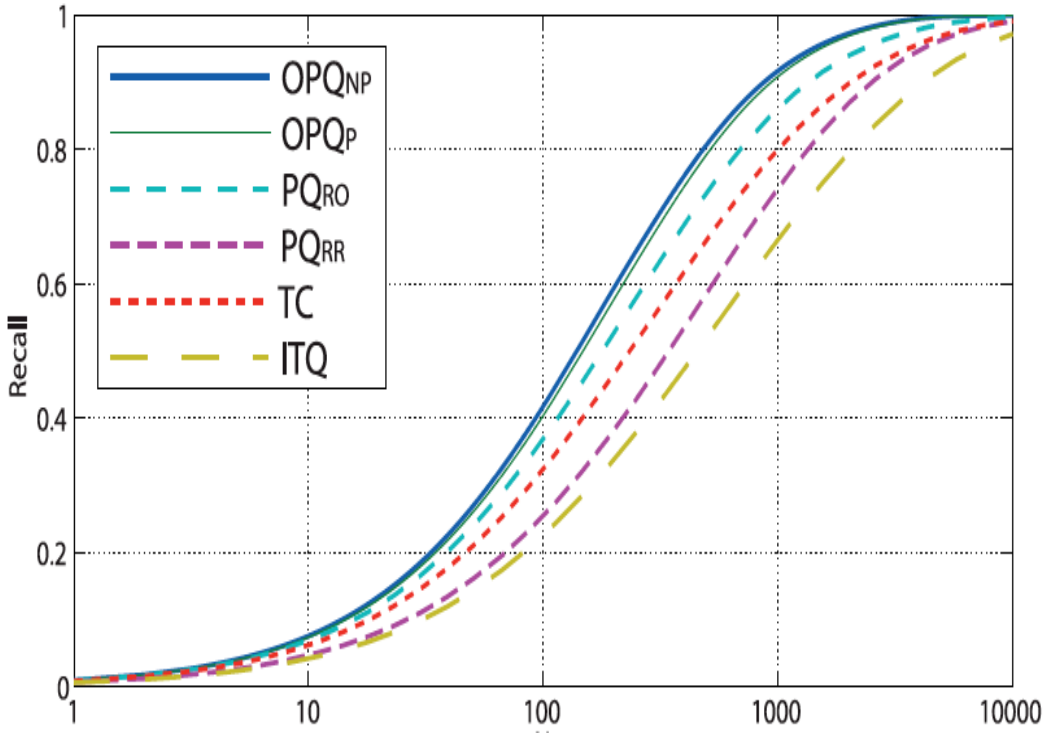
$$s.t. \quad \mathbf{c} \in \mathcal{C} = \{\mathbf{c} \,|\, R\mathbf{c} \in \mathcal{C}^1 \times \cdots \times \mathcal{C}^M, R^{\mathrm{T}}R = I\}$$

- Solutions
  - Nonparametric solution
    - (step1)Fix *R*, estimate clusters **c** and assignment *i*(x) – k-means
    - (step2)Fix *C*'s and optimize *R* – *orthogonal procrustes problem*
    - Alternate step1 and step2 until max iteration
  - Parametric solution
    - Assumes Gaussian distribution
    - Eigenvalue allocation algorithm
      - Align the data by PCA (make subspaces independent)
      - Allocate eigenvalues to buckets with balance

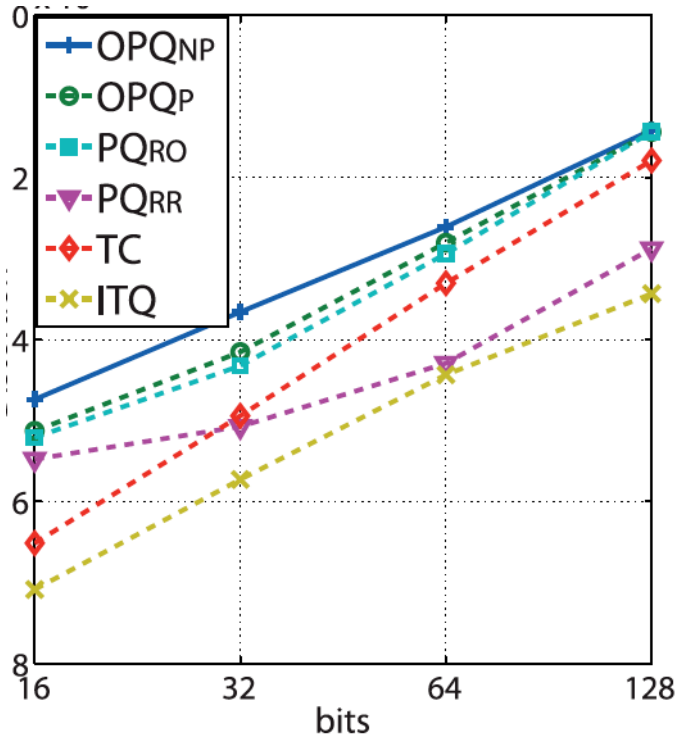
Optimized Product Quantization

- Results

[Ge et al., CVPR 2013]
[Norozi and Fleet, CVPR 2013]



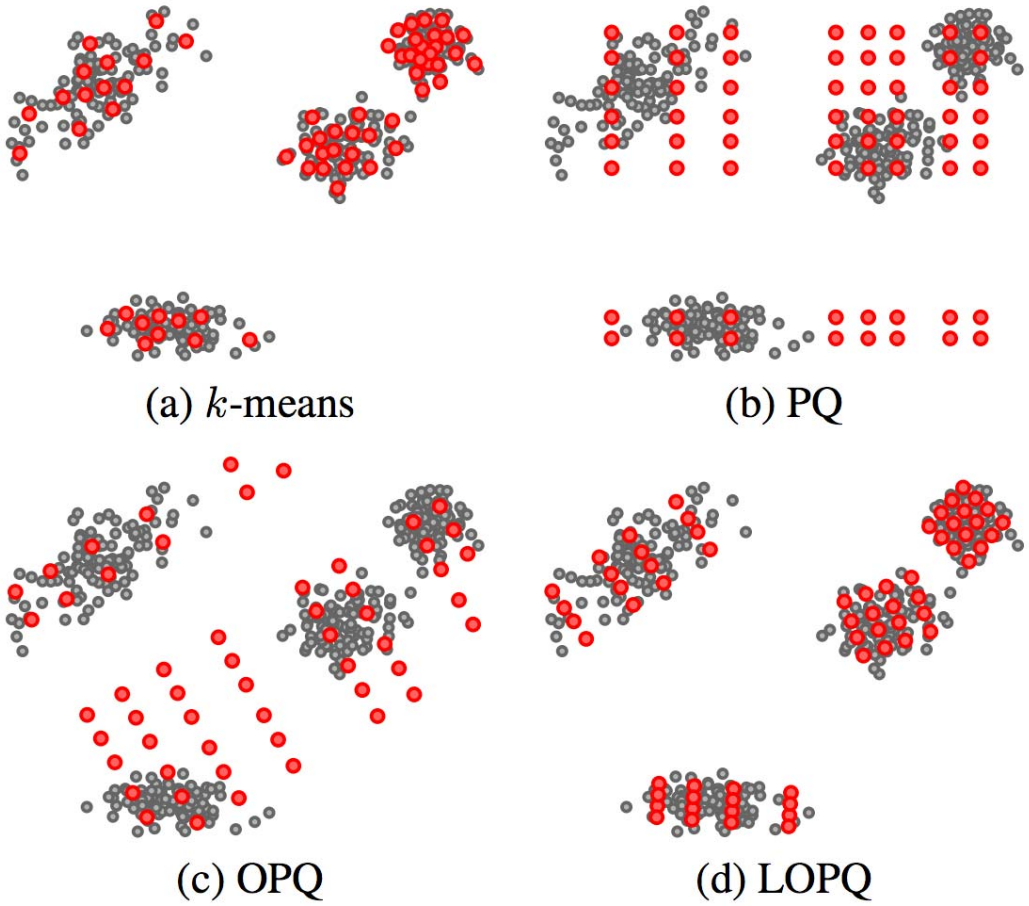
SIFT 1M (64bit code ADC)



Distortion vs Codelength

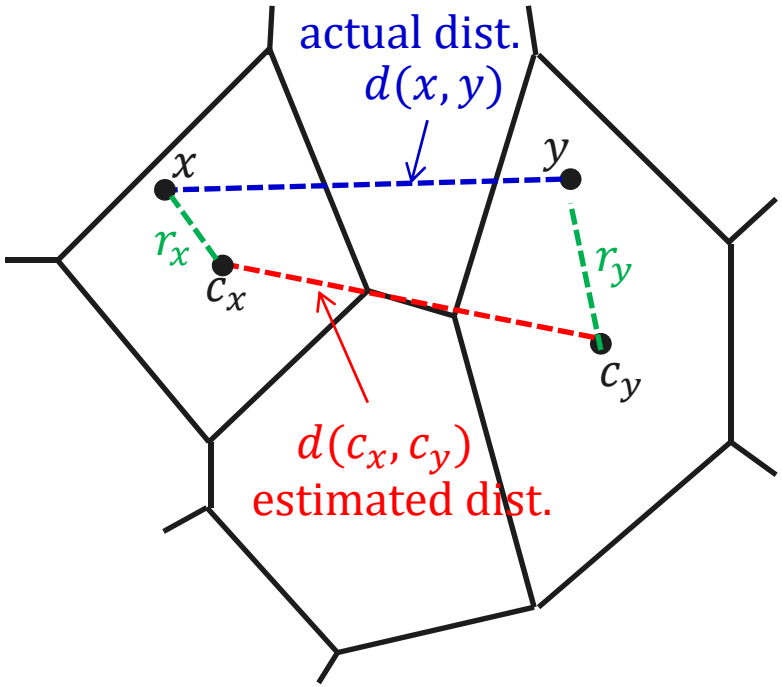
Locally Optimized PQ

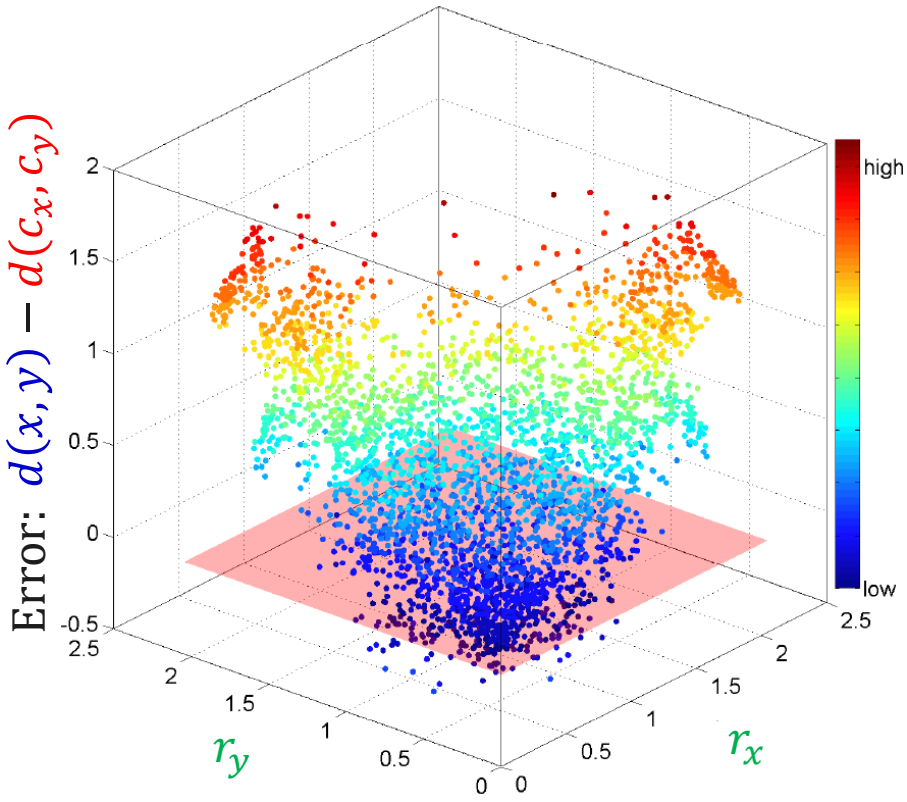
[Kalantidis et al., CVPR 2014]



Distance Encoded PQ

[J. Heo et al., CVPR 2014]

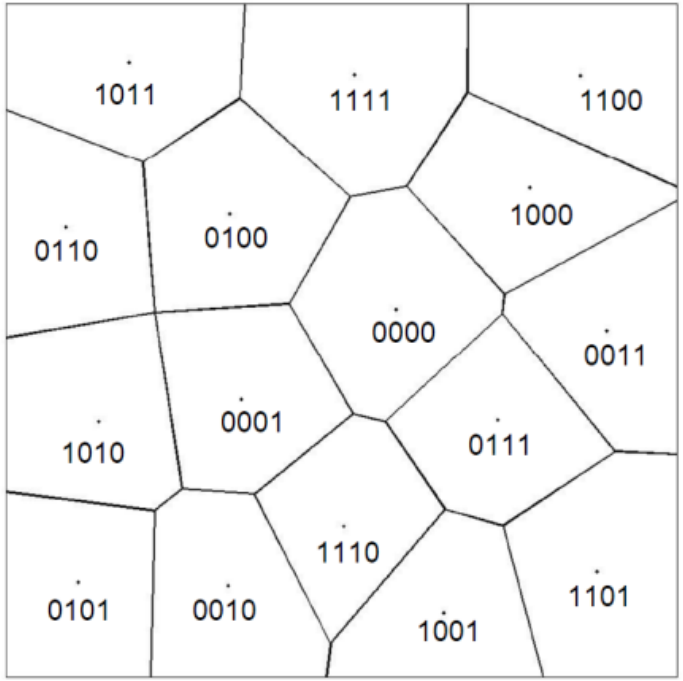




In PQ, errors of estimated distances $d(x,y) - d(c_x,c_y)$ tends to be higher as r_x and r_y becomes larger.

Distance Encoded PQ (DPQ)

- Encode quantized distance from cluster center as well as the cluster index

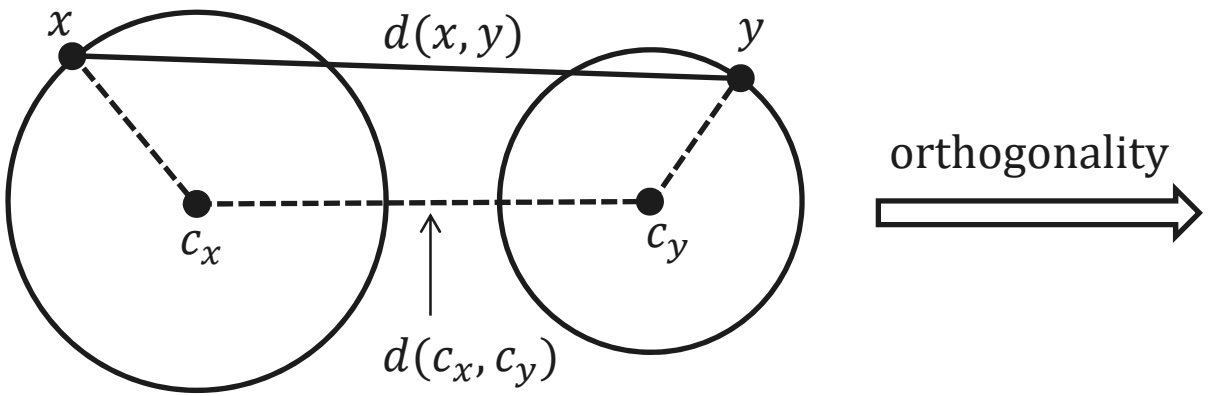


PQ

DPQ

Orthogonality in High Dim. Space

In high dimensional space, two randomly chosen vectors are highly likely to be orthogonal*.

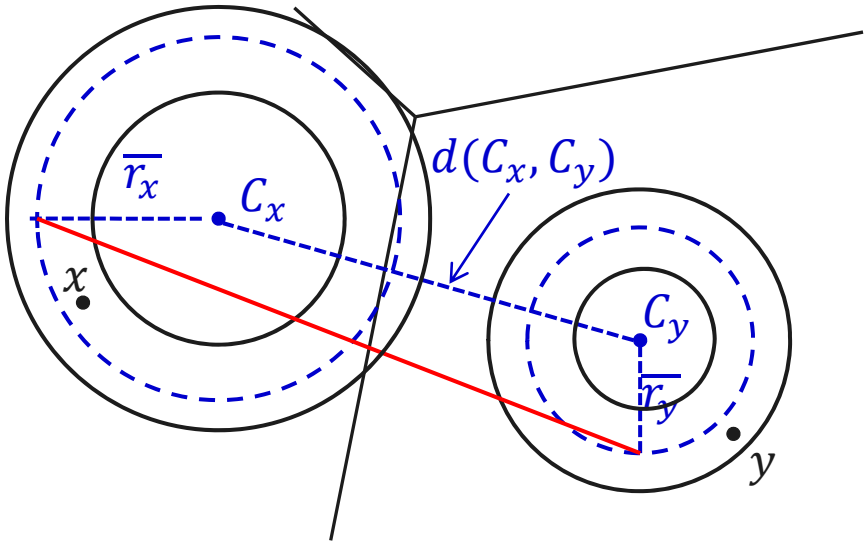


$$d(x,y)^2 = \|x-y\|^2 = \|(c_x - c_y) + (x - c_x) + (y - c_y)\|^2$$
$$\approx \|c_x - c_y\|^2 + \|x - c_x\|^2 + \|y - c_y\|^2$$

($\because$ $c_x - c_y, x - c_x$, and $y - c_y$ are mutually orthogonal.)

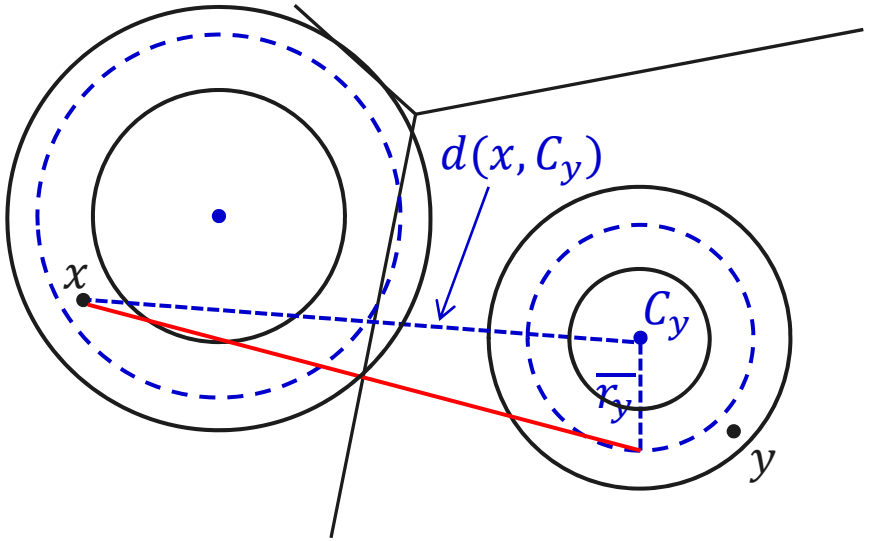
*Mathematical foundations for the information age [J. Hopcroft, 2010]

Distance Estimation



Symmetric Distance

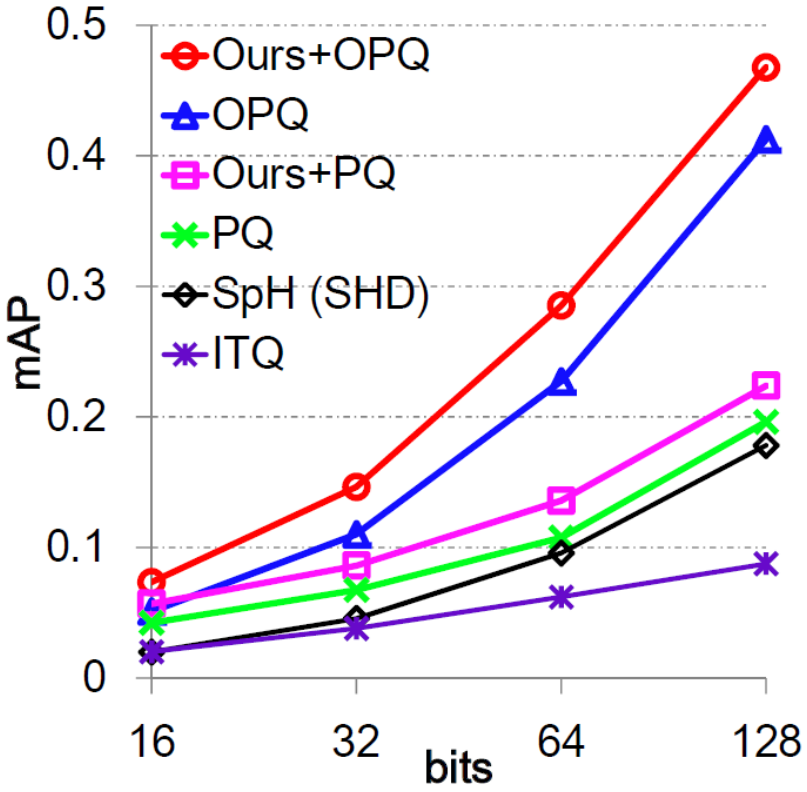
$$d_{SD}^{DPQ}(x,y)^2 = d(C_x,C_y)^2 + \overline{r_x}^2 + \overline{r_y}^2$$



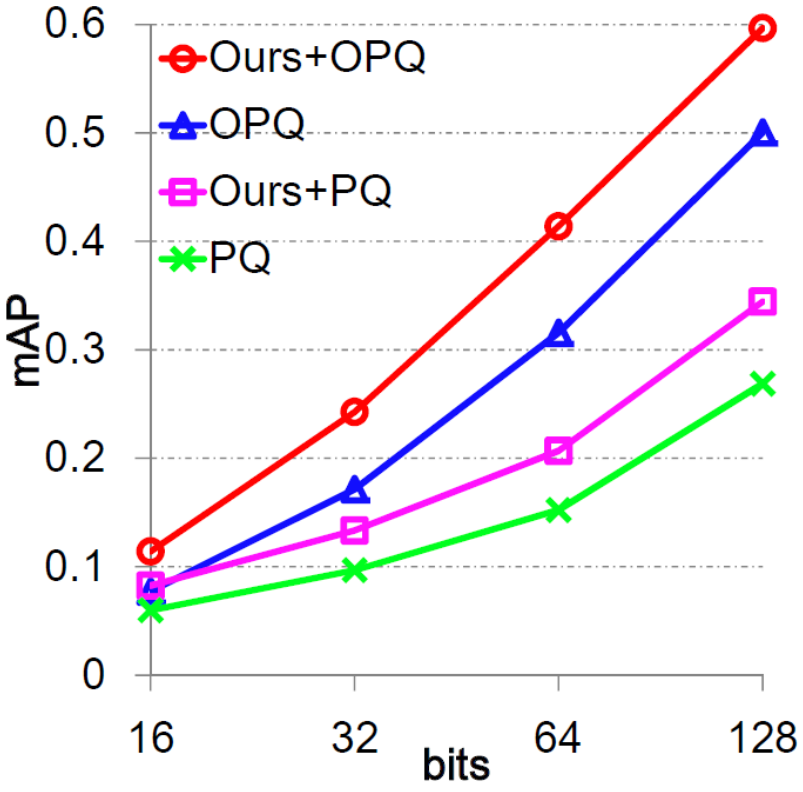
Asymmetric Distance

$$d_{AD}^{DPQ}(x,y)^2 = d(x,C_y)^2 + \overline{r_y}^2$$

Result (1M, 960-Dim GIST)



Symmetric distance



Asymmetric distance

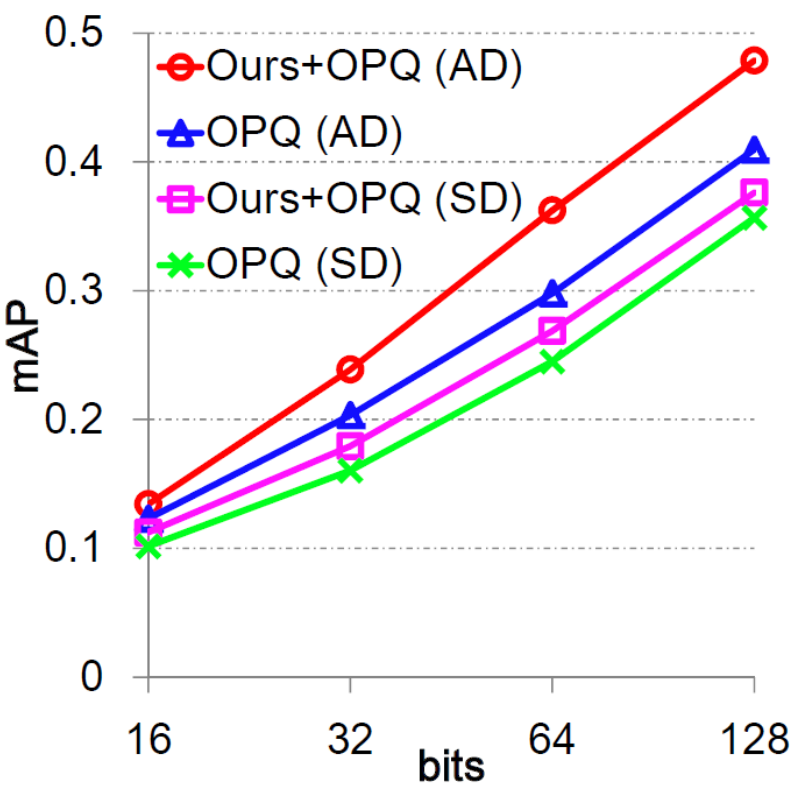
1000-nearest neighbor search mAP

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

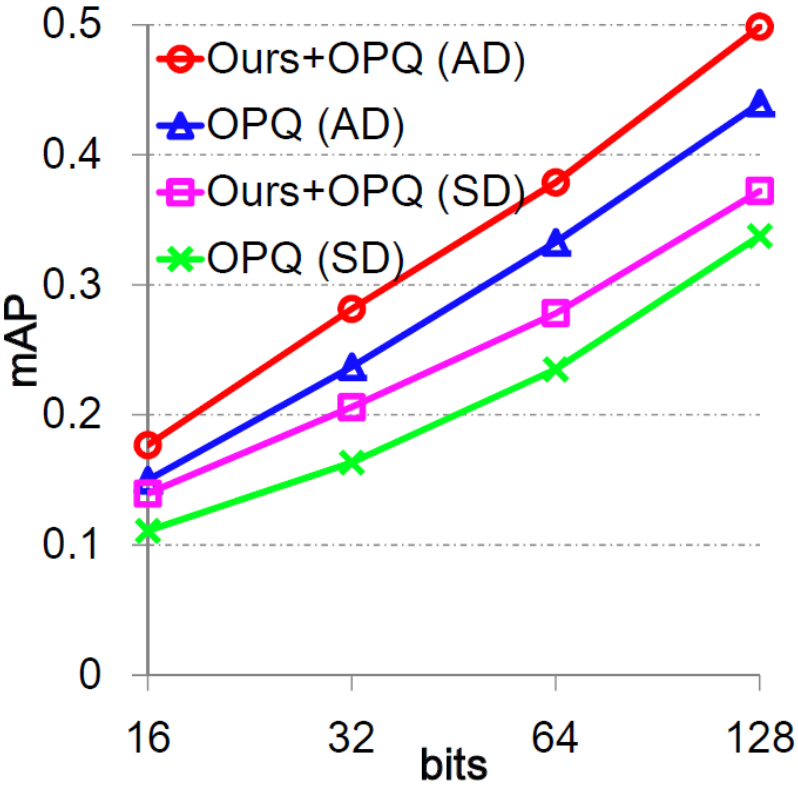
SpH: Sperical Hashing [Heo et al., CVPR 2012]

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

Result (1M, 1024-Dim BoW)



Original Data



L_2 Normalized data

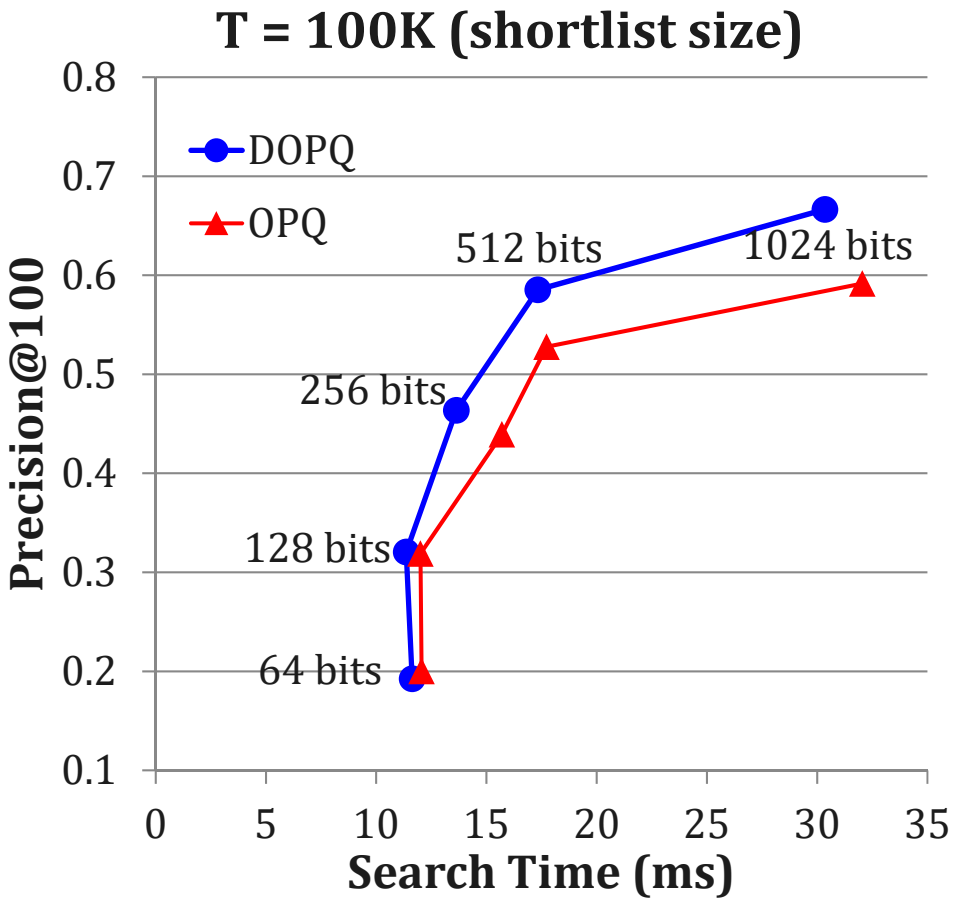
1000-nearest neighbor search mAP

SD: Symmetric distance

AD: Asymmetric distance

Result (Accuracy/Time/Memory)

- Tested on 4096-dimensional 11M CNN features
- Indexer: Vector Quantization with 4K centroids (4K lists)



Indexing and Encoding

- **Encoding**
  - Residual vector as input to encoder
  - Product quantization or its variant

- **Indexing**
  - Inverted (multi-)index
  - Residual-aware shortlist selection

Inverted Index

Construction time:

- Generate a codebook by quantization
  - e.g. k-means clustering

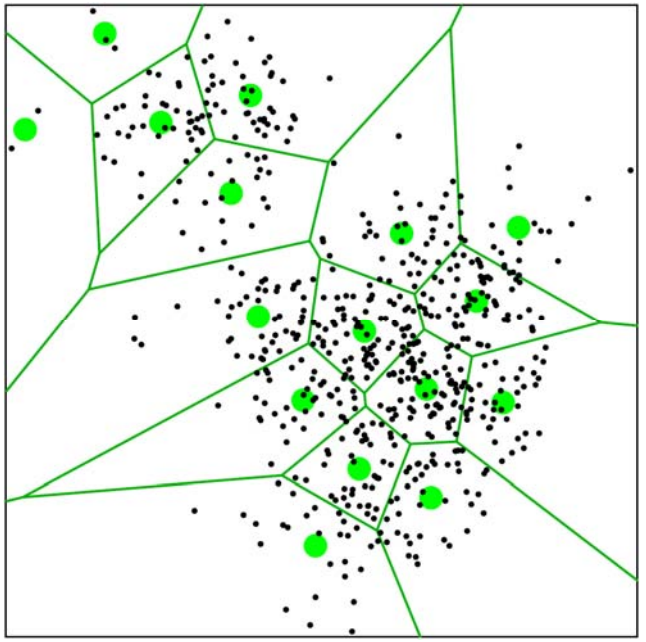
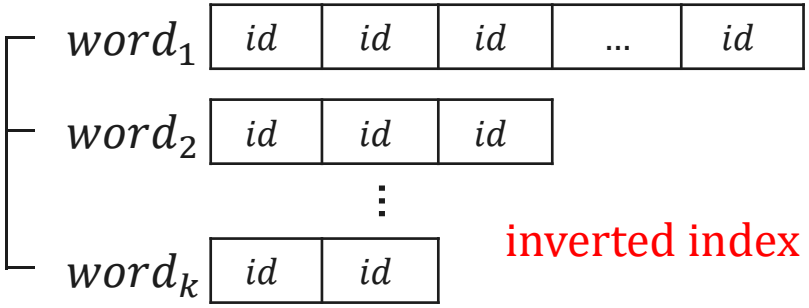


Figure from Lempitsky's slides

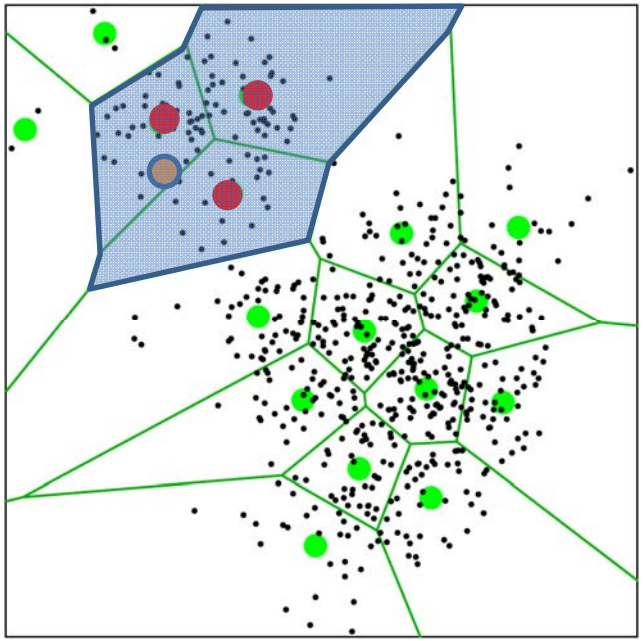
- Build an inverted index
  - Quantize each descriptor into the closest word
  - Organize desc. IDs in terms of words



Inverted Index

Query time:

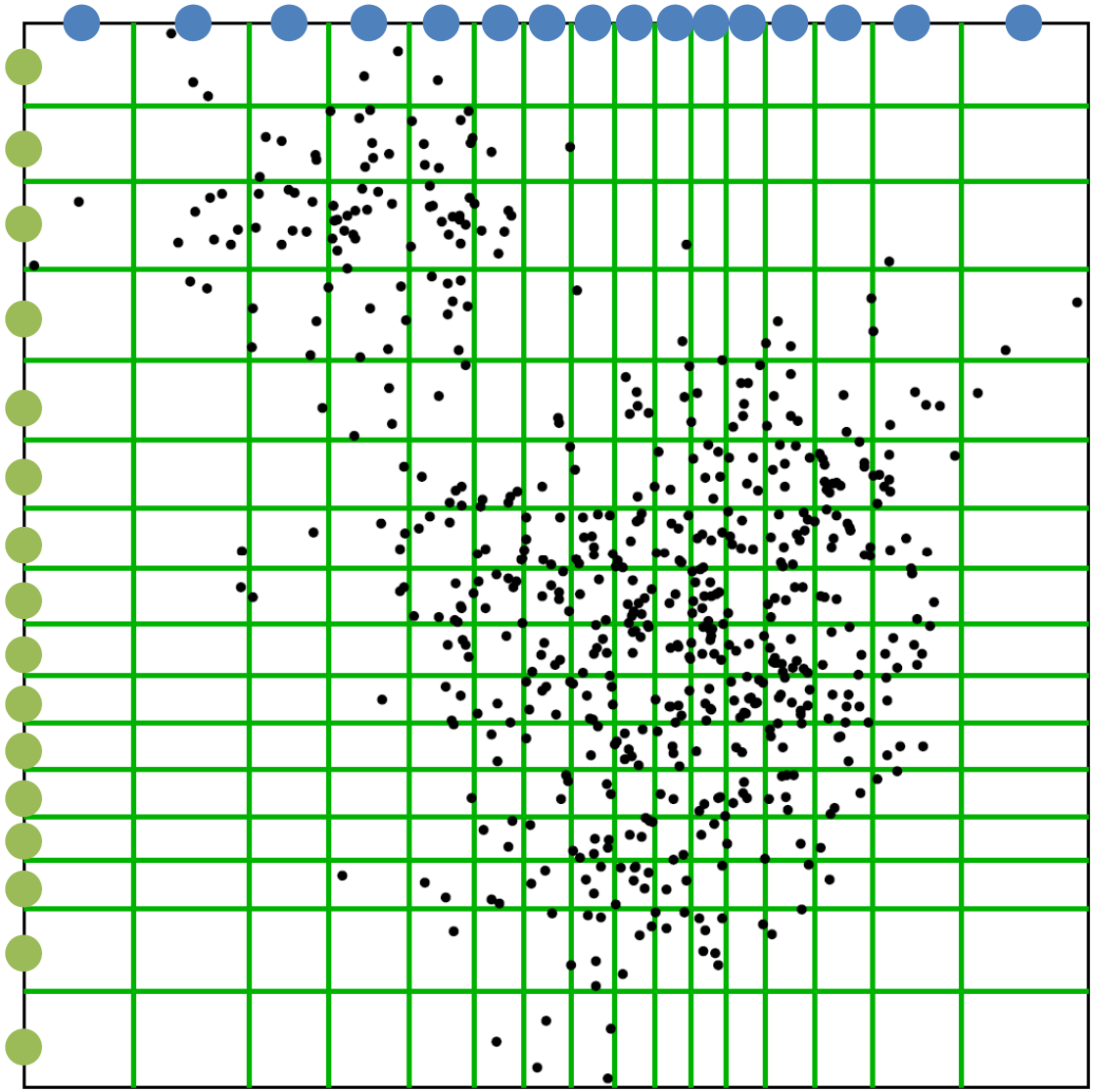
- Given a query,
  - Find its K closest words
  - Retrieve all the data in the K lists corresponding to the words
- Large K
  - Low quantization distortion
  - Expensive to find kNN words



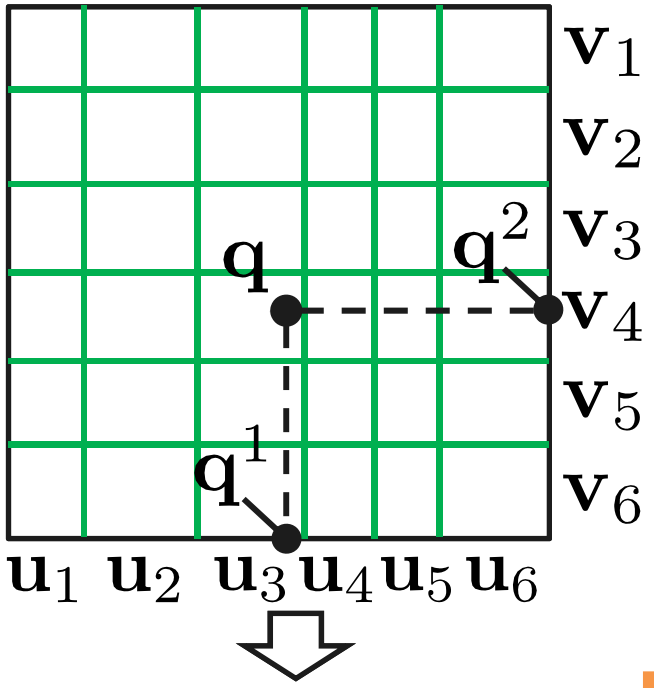
Inverted Multi-Index [Babenko and Lempitsky, CVPR 2012]

- **Product quantization for indexing**

- **Main advantage:**
  - For the same K, much finer subdivision
  - Very efficient in finding kNN codewords



Inverted Multi-Index [Babenko and Lempitsky, CVPR 2012]



$\mathbf{q}^2$ 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

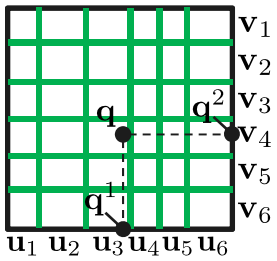
$\mathbf{q}^1$ vs. $\mathcal{U}$

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

	inverted index	inverted multi-index
number of entries	K	K^2
operations to match to codebooks	2K+O(1)	2K+O(1)

Inverted Multi-Index [Babenko and Lempitsky, CVPR 2012]

multi-sequence algorithm



$\mathbf{q}^1$ 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

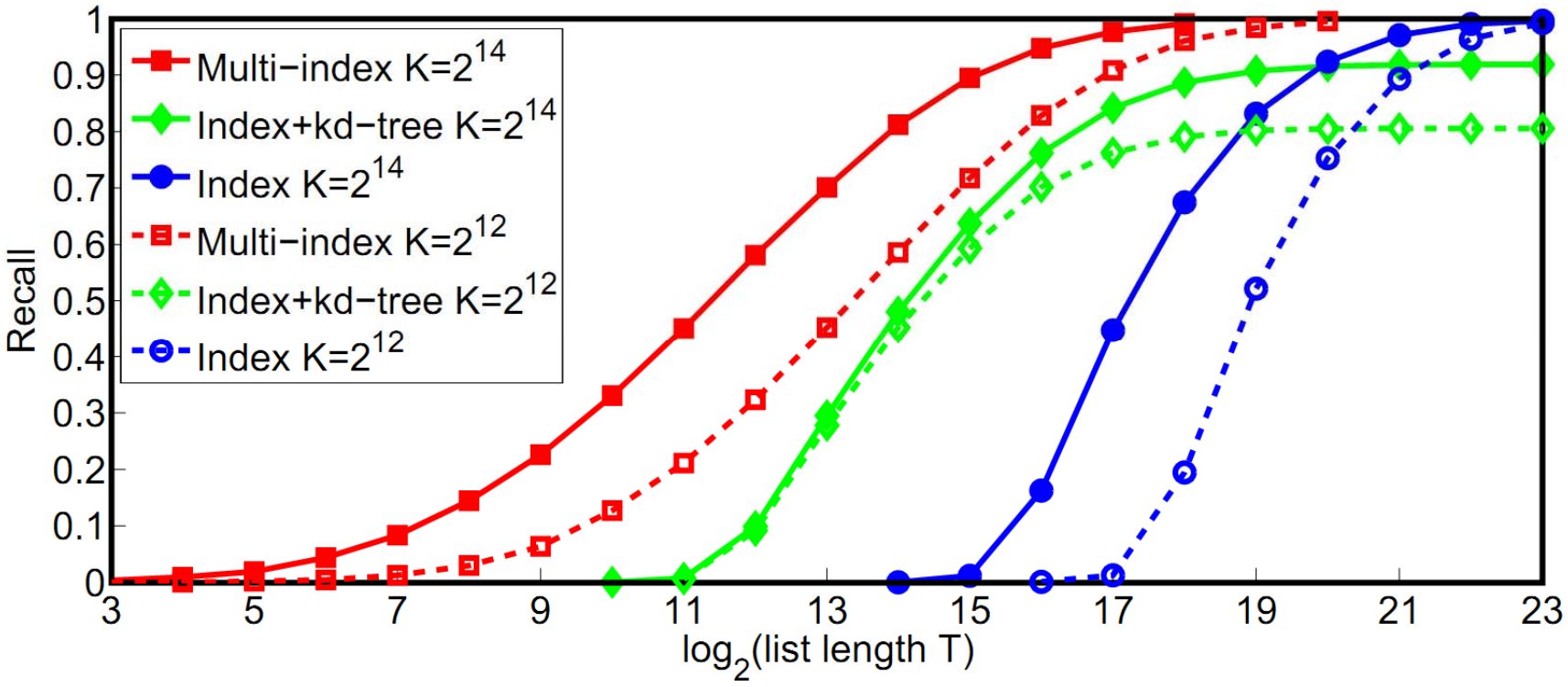
$[\mathbf{u}_{\alpha(i)}\ \mathbf{v}_{\beta(j)}]$	(i,j)	$r(i)+s(j)$
$[\mathbf{u_3}\ \mathbf{v_4}]$	(1,1)	0.6 (0.5+0.1)
$[\mathbf{u_4}\ \mathbf{v_4}]$	(2,1)	0.8 (0.7+0.1)
$[\mathbf{u_3}\ \mathbf{v_3}]$	(1,2)	2.5 (0.5+2)
$[\mathbf{u_4}\ \mathbf{v_3}]$	(2,2)	2.7 (0.7+2)
$[\mathbf{u_3}\ \mathbf{v_5}]$	(1,3)	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	$\mathbf{v}_1$
	$\mathbf{u}_3$	$\mathbf{u}_4$	$\mathbf{u}_5$	$\mathbf{u}_2$	$\mathbf{u}_1$	$\mathbf{u}_6$	

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

Inverted Multi-Index

[Babenko and Lempitsky, CVPR 2012]

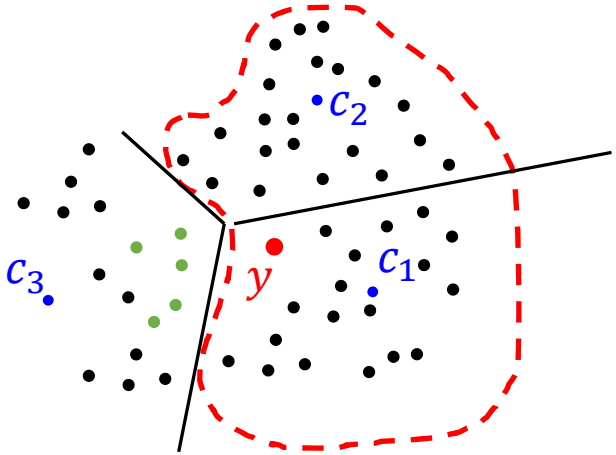


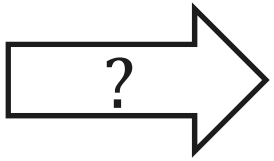
1B SIFT Dataset

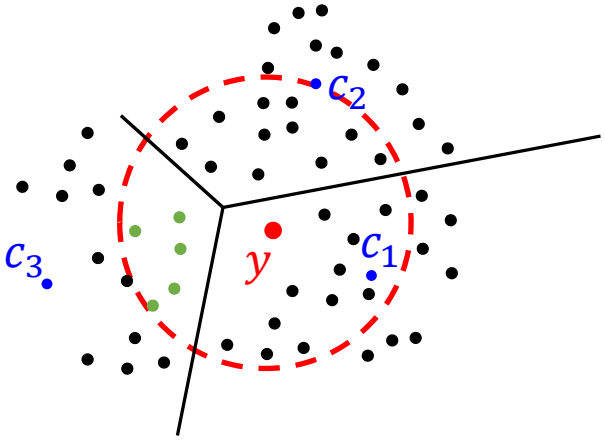
Residual-Aware Shortlist Retrieval

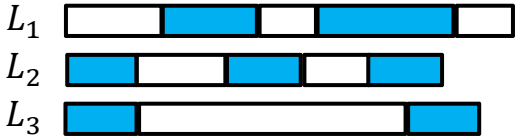
[Jaepil et al., CVPR 2016]

Limitation of prev. methods









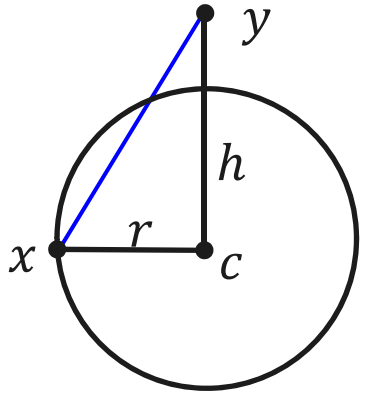
Neighbors could be missed due to the quantization error

Select promising subset in parallel from all the lists

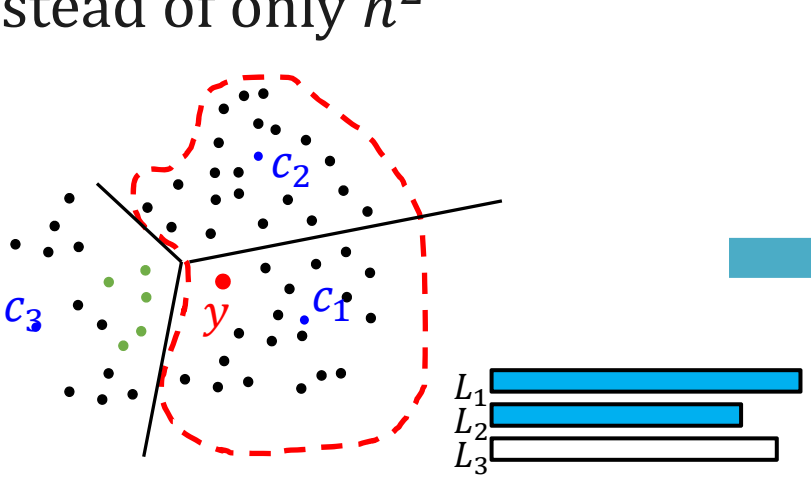
Motivation: Better Distance Estimator ?

- Distance estimator between y and x based on orthogonality in high-dimensional space:

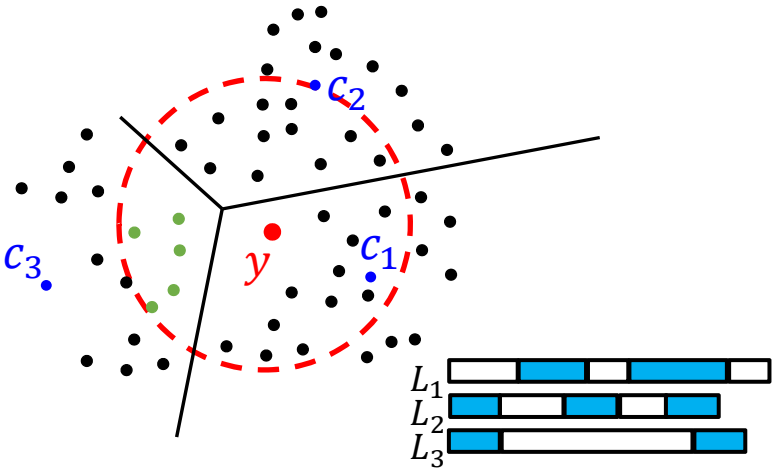
$$d(y,x)^2 \approx d(y,c)^2 + d(c,x)^2 = h^2 + r^2$$



- Compute shortlist according to $h^2 + r^2$, instead of only h^2



h^2



$h^2 + r^2$

Distance Estimator

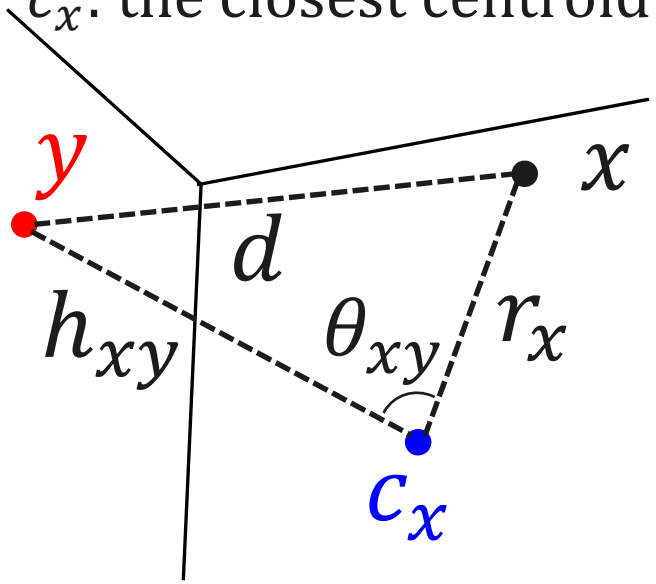
$$d^2 = h_{xy}^2 + r_x^2 - 2h_{xy}r_x\cos\theta_{xy}$$

$$= h_{xy}^2 + \left(1 - \frac{2h_{xy}}{r_x}\cos\theta_{xy}\right)r_x^2$$

$$\approx h_{xy}^2 + \alpha_K r_x^2$$

y: query / x: data

c_x: the closest centroid of x

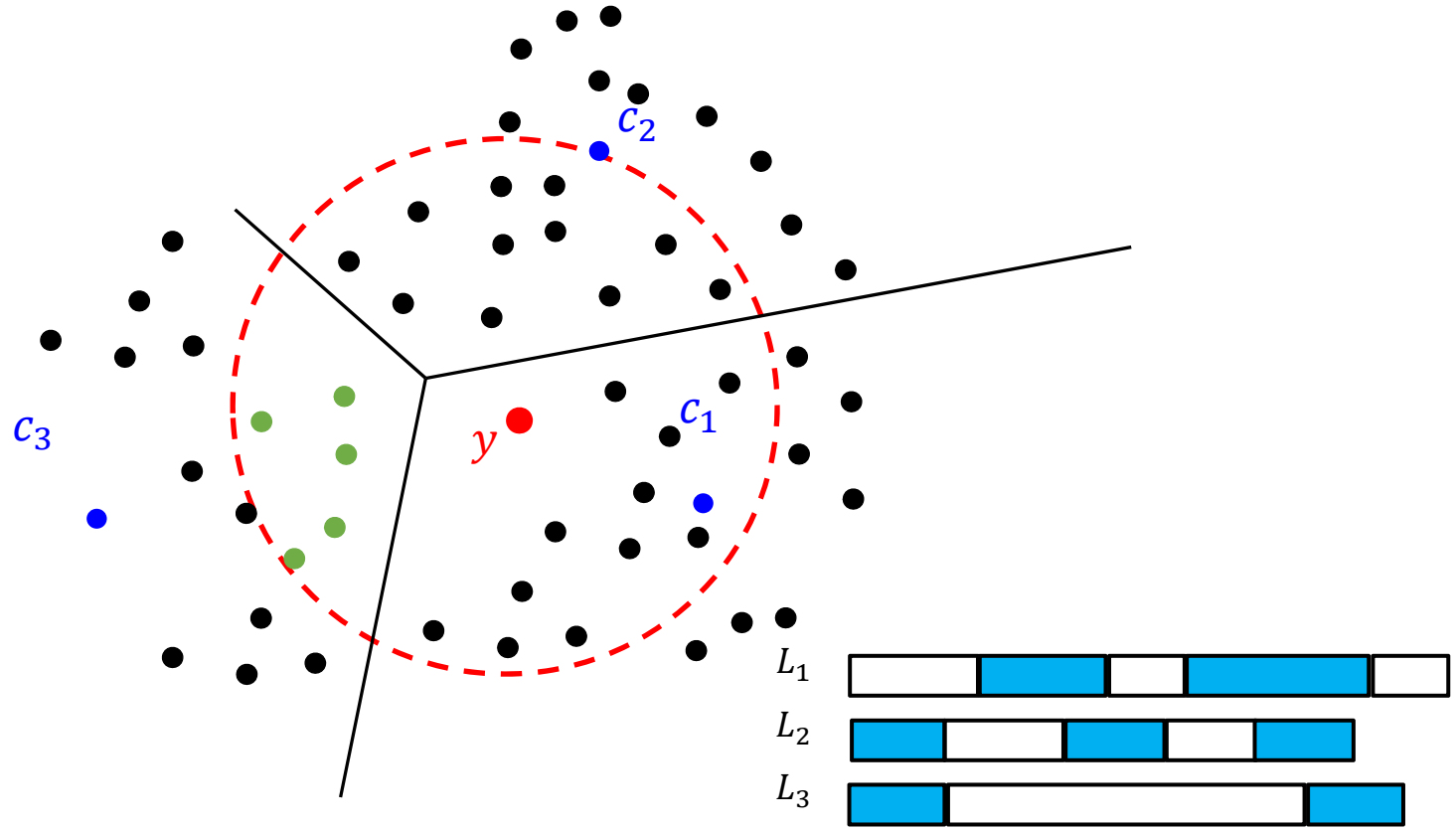


α_K is a constant depending on K (= number of desired neighbors)

= avg([ ] with K-NN pairs and [ ] with random pairs)

e.g) α_1= 0.5 / α_{10}= 0.55 / α_{100}= 0.62 / α_{1000}=0.70

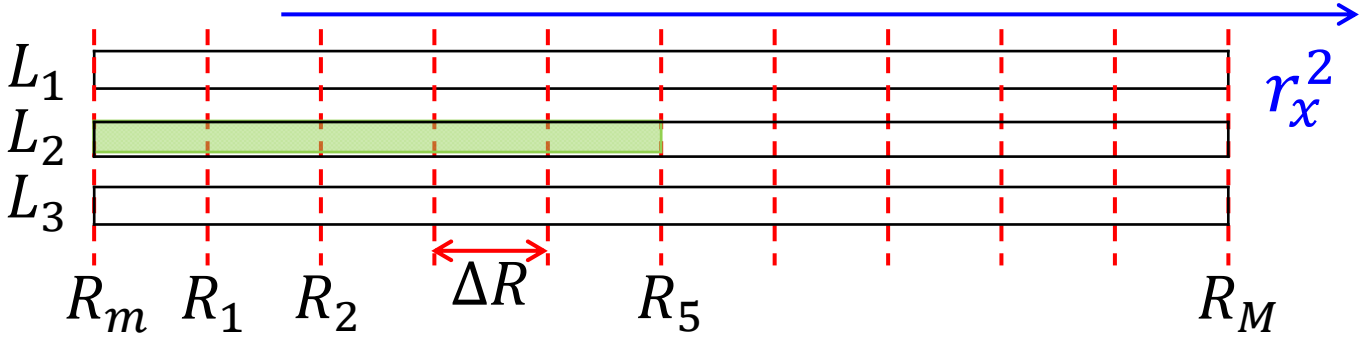
Distance Estimator



$$h^2 + \alpha_K r^2$$

Shortlist with Inverted Index

- **Lookup Table Precomputation**
  - Sort and partition each inverted list according to r_x^2



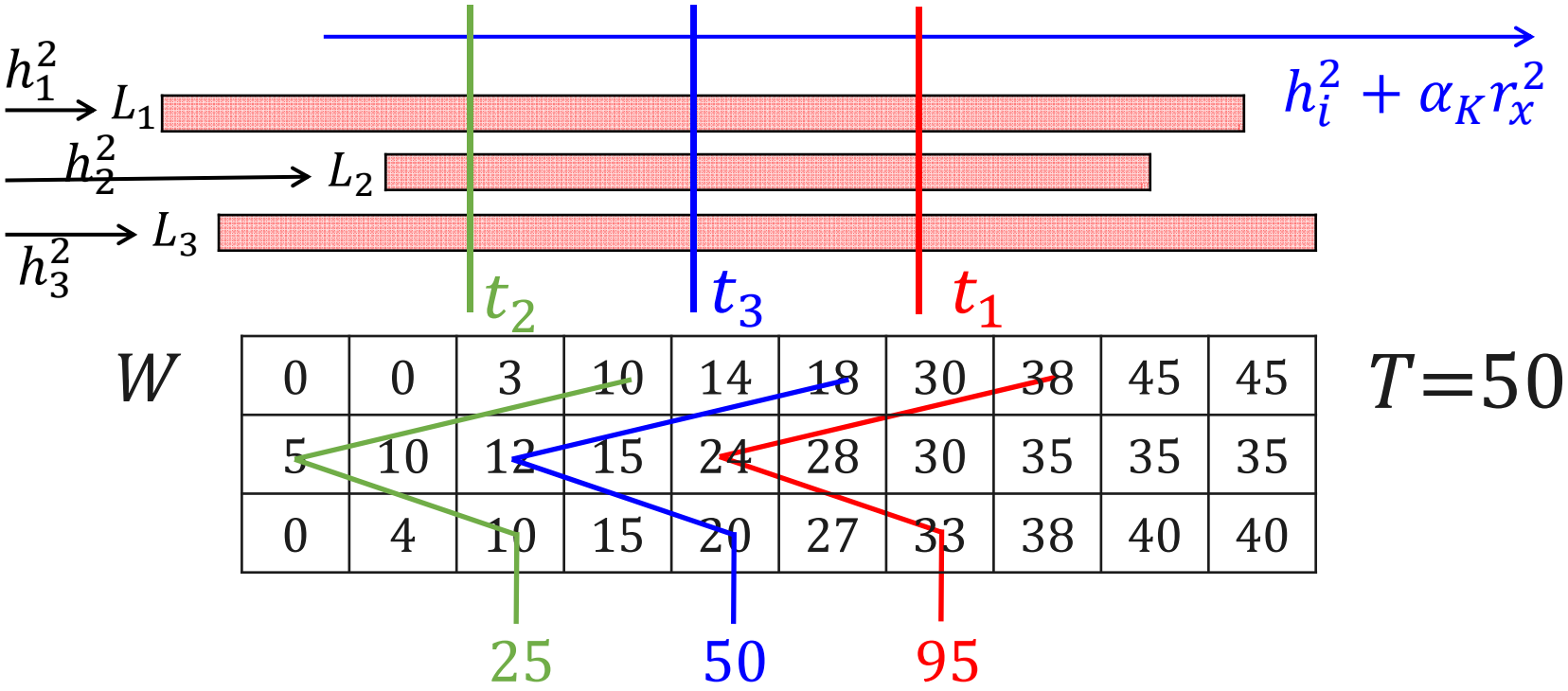
  - Compute a lookup table $W(i, j)$ – the number of data in L_i whose r_x^2 are smaller than R_j

W

0	0	3	10	14	18	30	38	45	45
5	10	12	15	24	28	30	35	35	35
0	4	10	15	20	27	33	38	40	40

Shortlist with Inverted Index

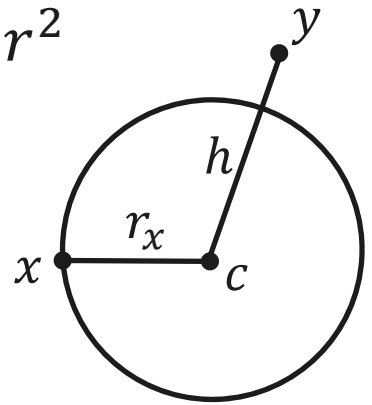
- **Runtime Shortlist Selection**
  - Given a query y, shortlist size T, and the target number of neighbors K
  - Compute squared distances to centroids h_i^2
  - Do binary search to find a distance threshold that corresponds to T



Shortlist with Inverted Multi-Index

- **Residual-Aware Indexing**
  - Partition each cluster by r_x to define index (subspace ID, cluster ID, distance ID)
  - Compute a representative residual distance $\bar{r}^2_{s,i,j}$ for each index (s, i, j)

$h^2 + \alpha_K r^2$



Subspace #1

Subspace ID

Subspace #2	(1,1,1)	(1,1,2)	(1,2,1)	(1,2,2)
(2,1,1)				
(2,1,2)				
(2,2,1)				
(2,2,2)				

Cluster ID

Distance ID

Subspace #1	Subspace #2
$\bar{r}^2_{1,1,1}$=0.3	$\bar{r}^2_{2,1,1}$=0.7
$\bar{r}^2_{1,1,2}$=1.3	$\bar{r}^2_{2,1,2}$=0.9
$\bar{r}^2_{1,2,1}$=0.4	$\bar{r}^2_{2,2,1}$=0.6
$\bar{r}^2_{1,2,2}$=0.8	$\bar{r}^2_{2,2,2}$=1.0

Shortlist with Inverted Multi-Index

- **Runtime Shortlist Selection**
  - Compute $h_{s,i}^2$ (the distance to i-th centroid of s-th subspace), and sort the indices in each subspace according to $h_{s,i}^2$
  - Traverse the table by using the multi-sequence algorithm

Subspace #1

$\bar{h}_{1,1}^2$=0.8
$\bar{h}_{1,2}^2$=1.0

Subspace #2

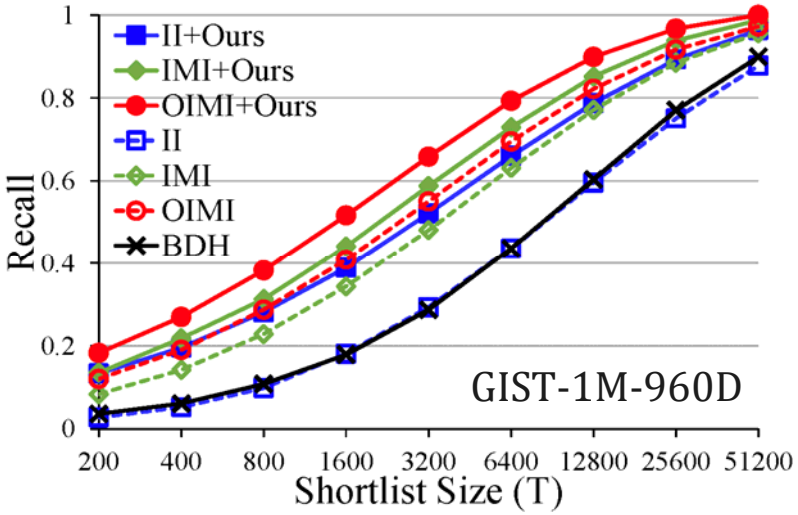
$\bar{h}_{2,1}^2$=1.1
$\bar{h}_{2,2}^2$=0.6

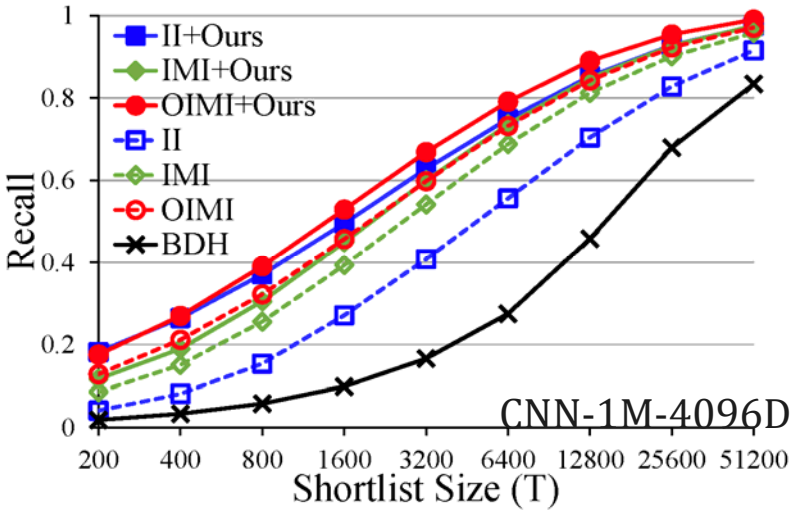
$h_{1,i}^2 + \alpha_{K,1} r_{1,i,j}^2$ →

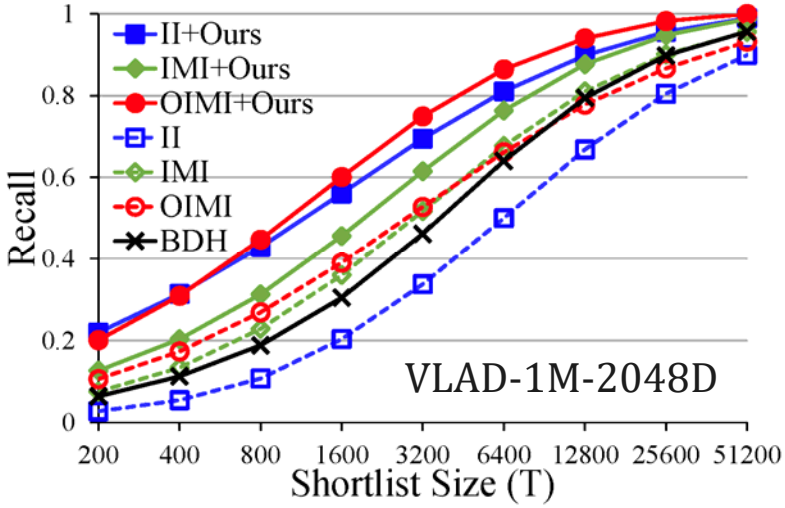
	(1,1,1) 1.1	(1,2,1) 1.4	(1,2,2) 1.8	(1,1,2) 2.1
(2,2,1) 1.2	2.3	2.6	3.0	3.3
(2,2,2) 1.6	2.7	3.0	3.4	3.7
(2,1,1) 1.8	2.9	3.2	3.6	3.9
(2,1,2) 2.0	3.1	3.4	3.8	4.1

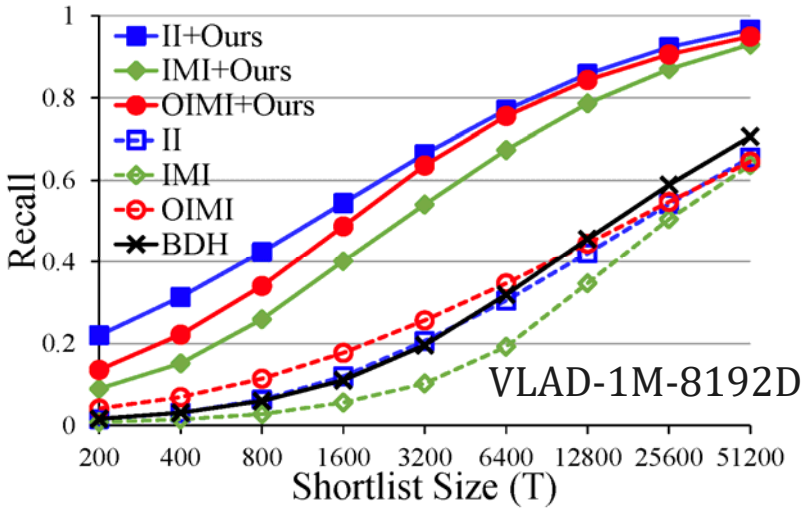
Shortlist Evaluation

Accuracy of Shortlist retrieving top 100 GT



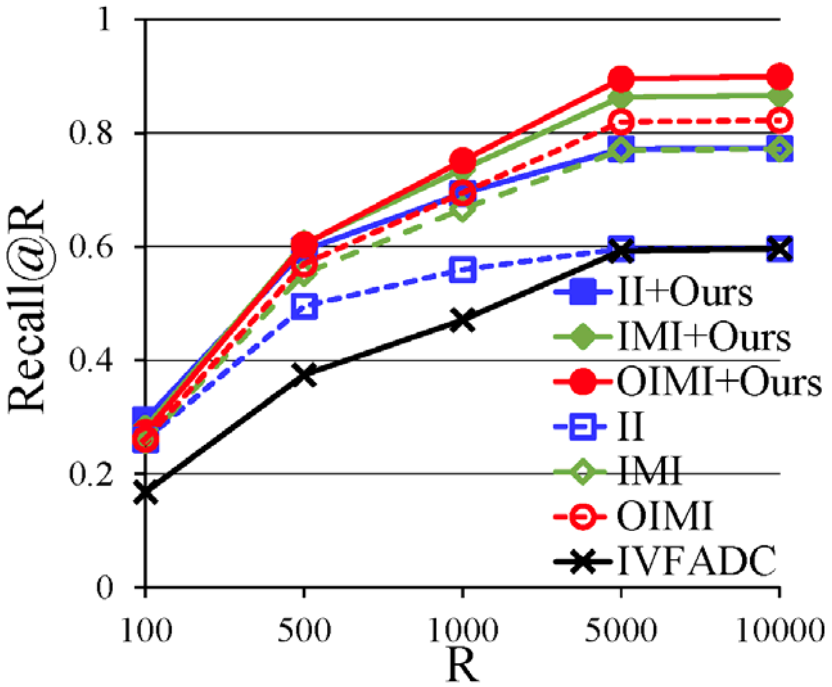




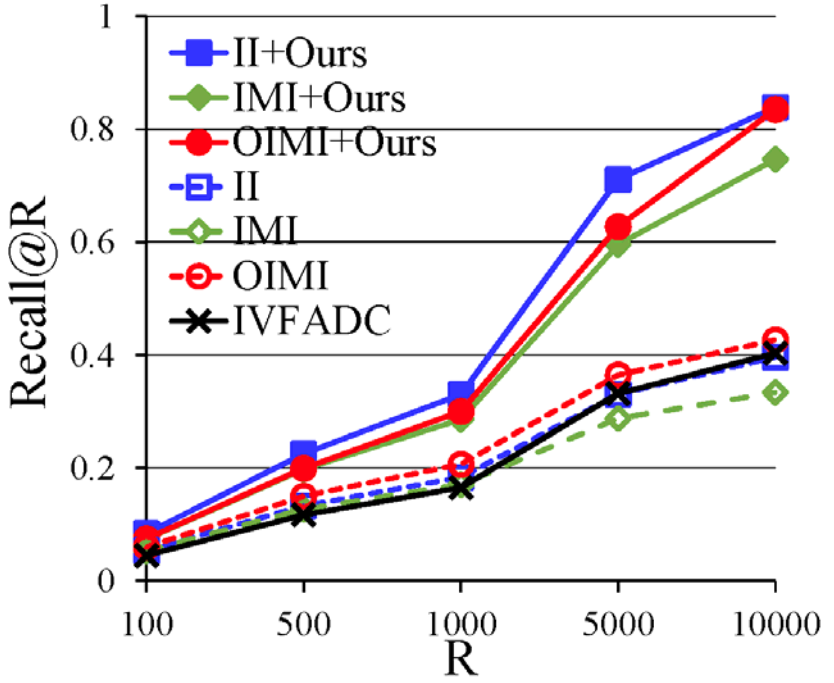


Shortlist Evaluation

End-to-End Accuracy



GIST-1M-960D, Re-ranked



VLAD-1M-8192D, Re-ranked

Applications

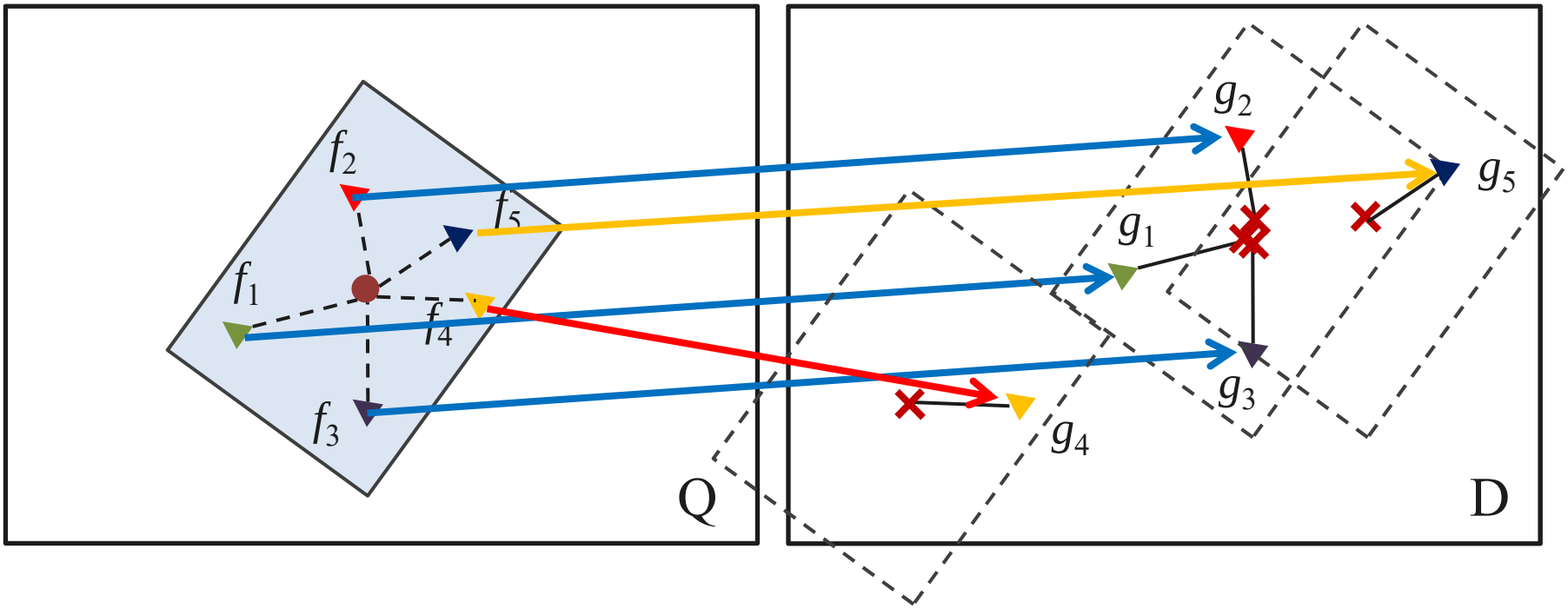
- Object retrieval and localization
- Product image recognition
- Face detection and attribute recognition
- Large-scale semantic image search
- Large-scale image tagging
- Free-text image search

Object Retrieval and Localization

[X. Shen et al., CVPR 2012]

Object Retrieval and Localization

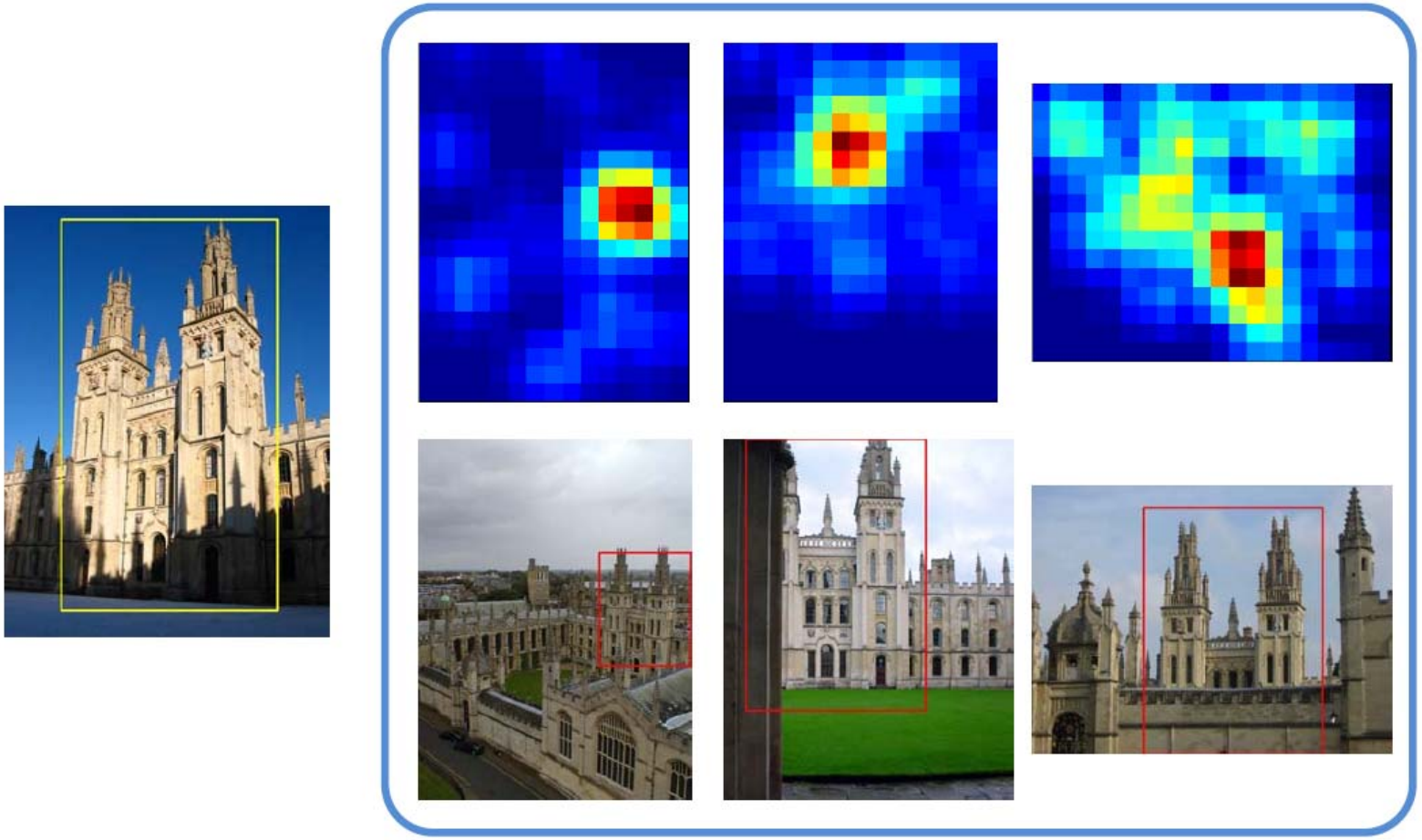
- Local correspondence voting for non-rigid object matching



tf-idf pair voting score: $\frac{idf(k) \cdot idf(k)}{tf(Q,k) \cdot tf(D,k)}$

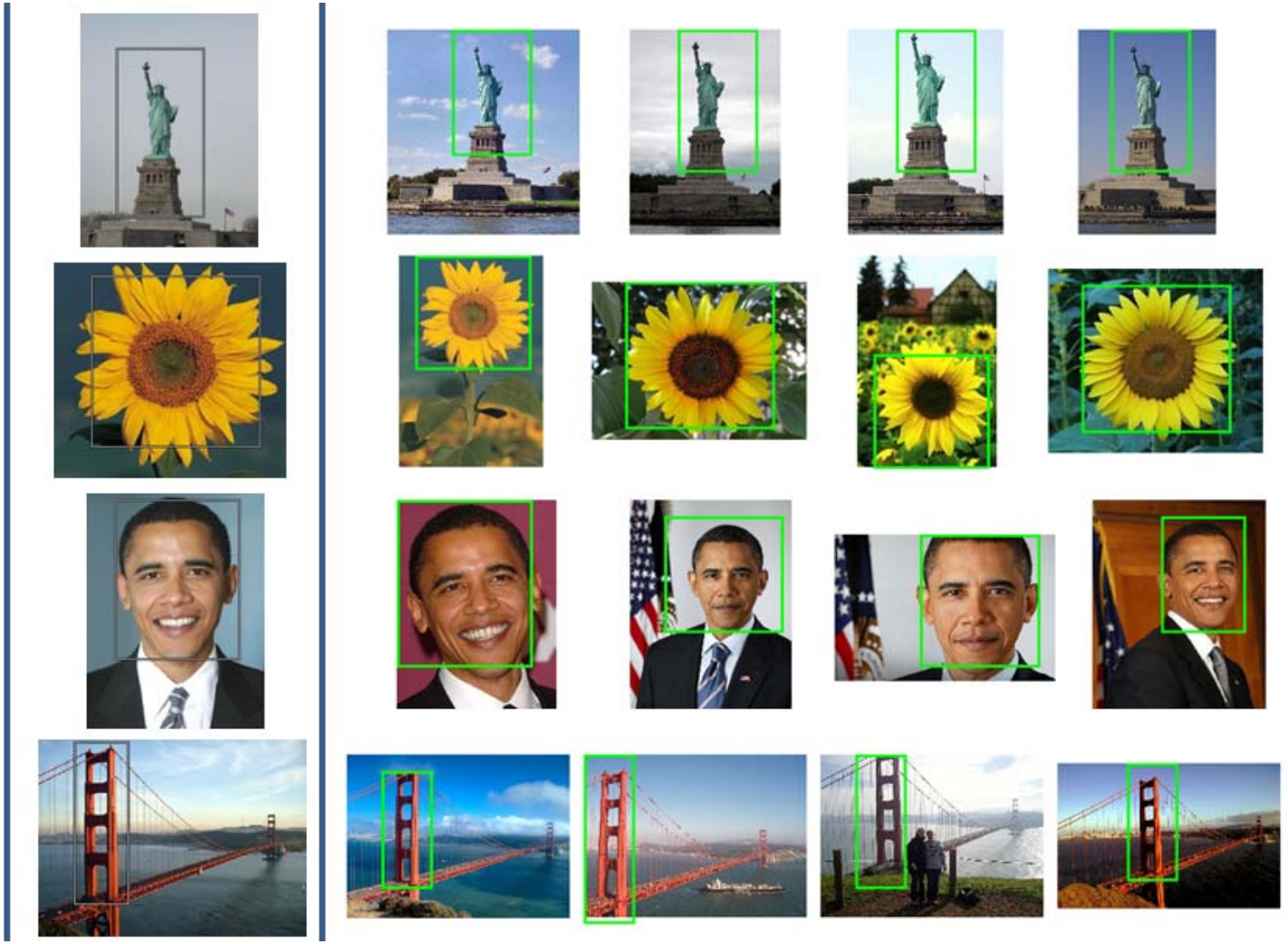
Choose the transformation with the highest score!

Object Retrieval and Localization



Examples of Voting Maps

Object Retrieval and Localization



Non-rigid cases

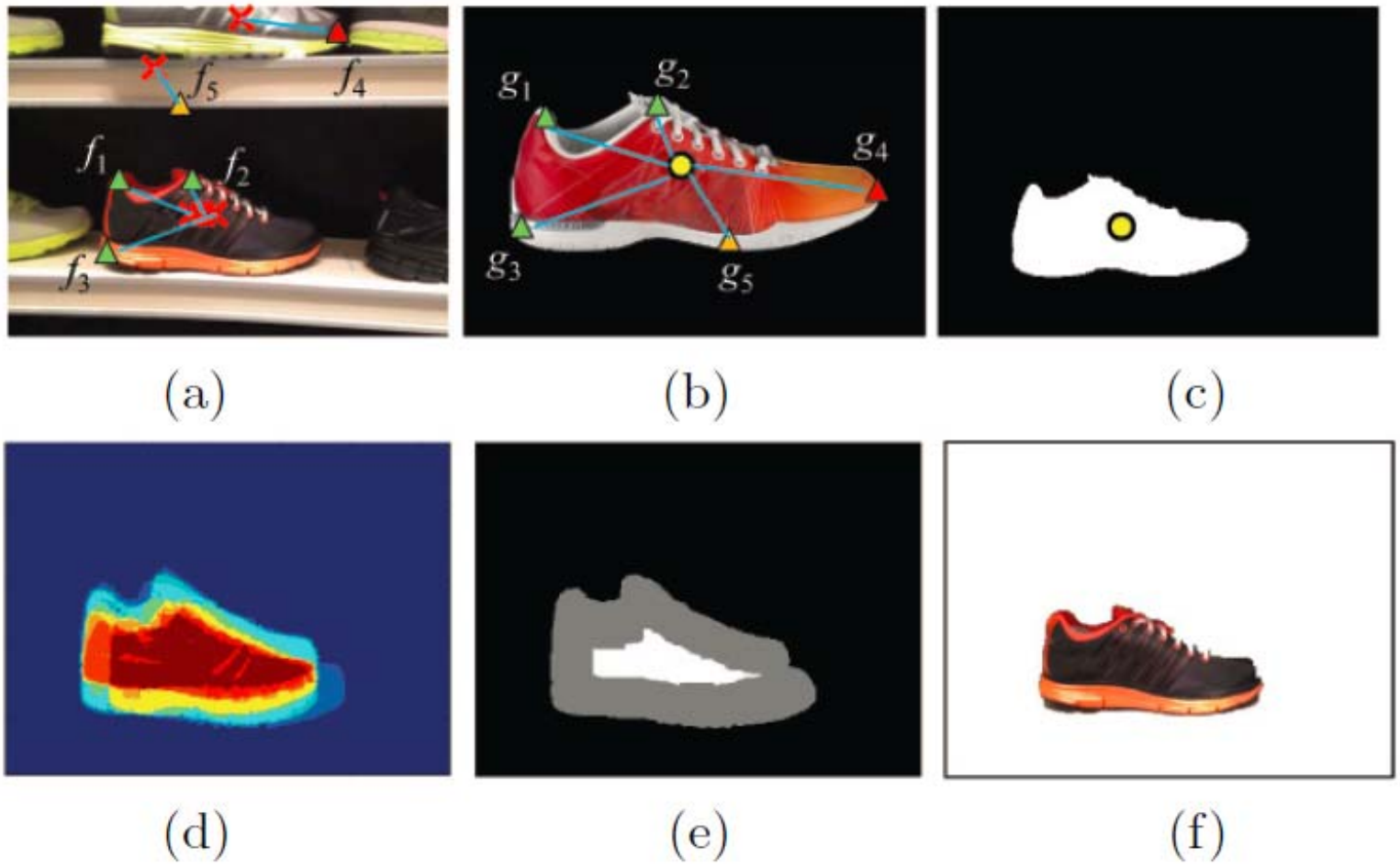
Product Image Recognition

[X. Shen et al., ECCV 2012]

Examples of product images in the database

Examples of query images taken by mobile phones

Product Image Recognition



Product Image Recognition

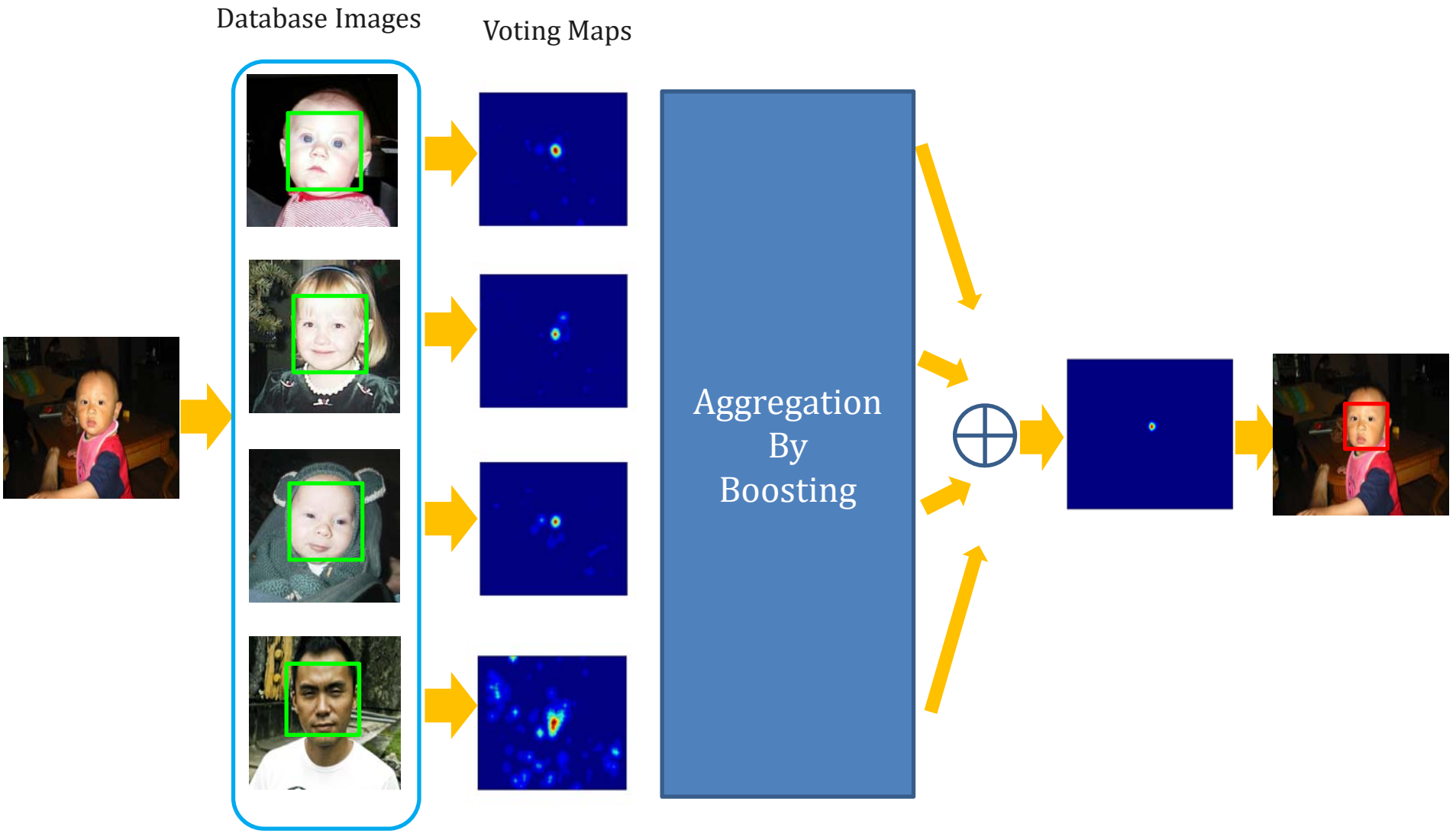
Images Support map Extraction GrabCut

Face Detection by Image Retrieval

[X. Shen et al., CVPR 2013]

[H. Li et al., CVPR 2014]

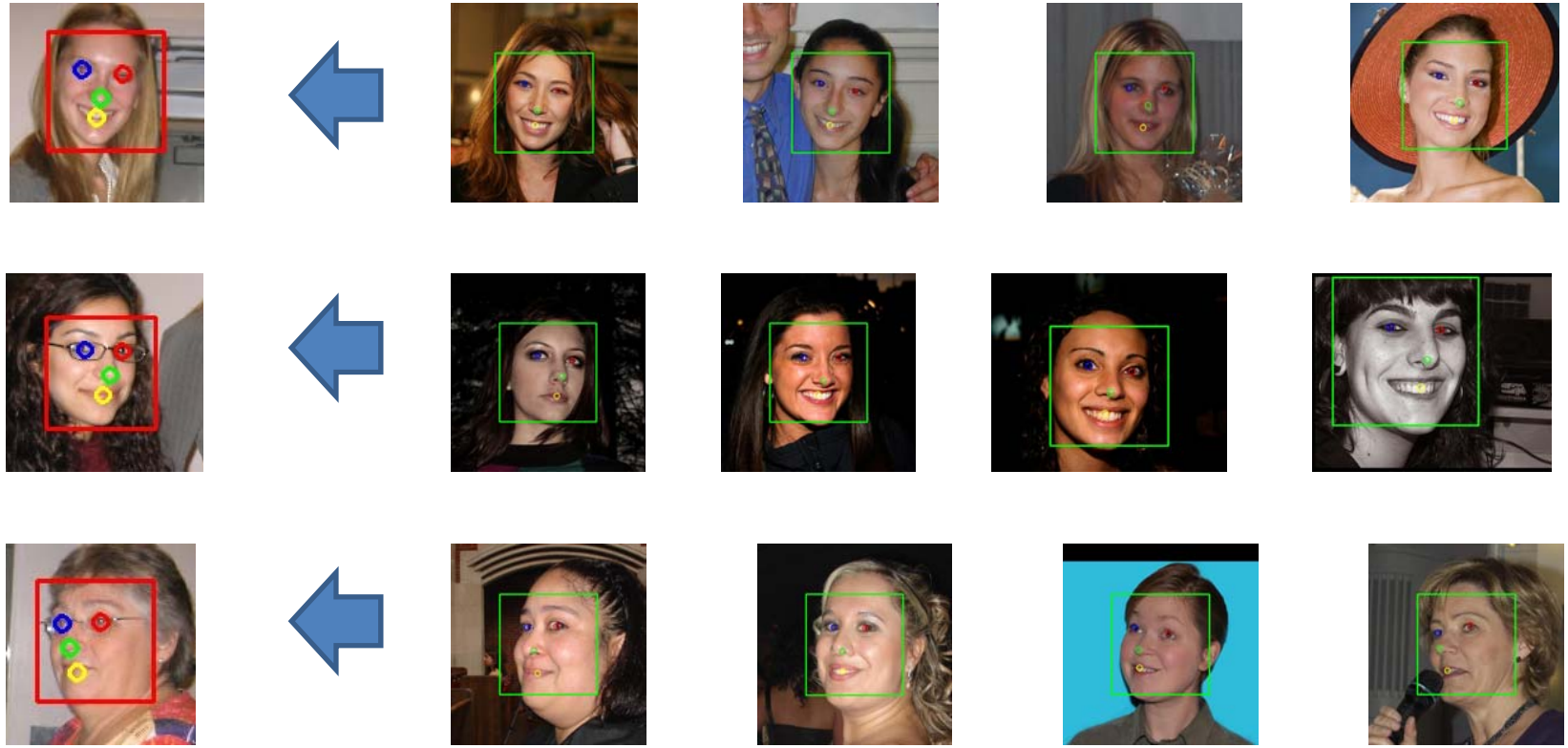
Face Detection by Image Retrieval



Face Detection by Image Retrieval

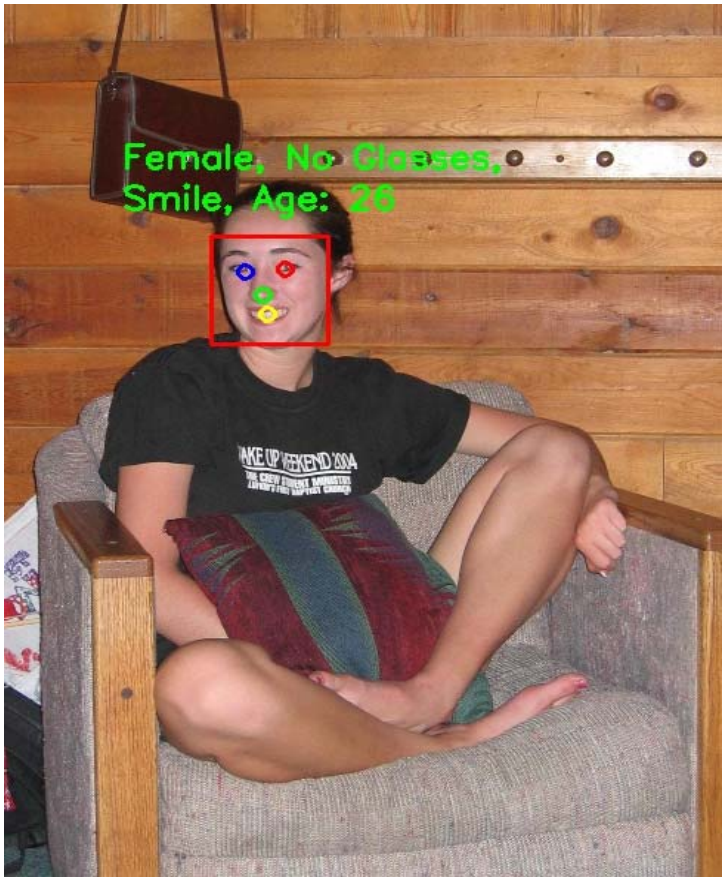
Example detection results

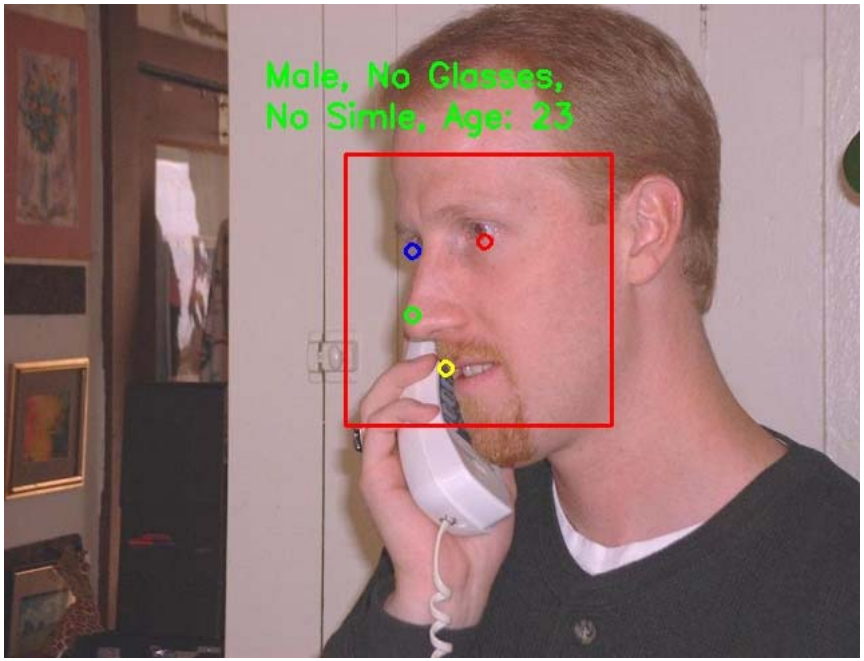
Facial Attribute Recognition



transfer landmark, pose, age, gender, expression…

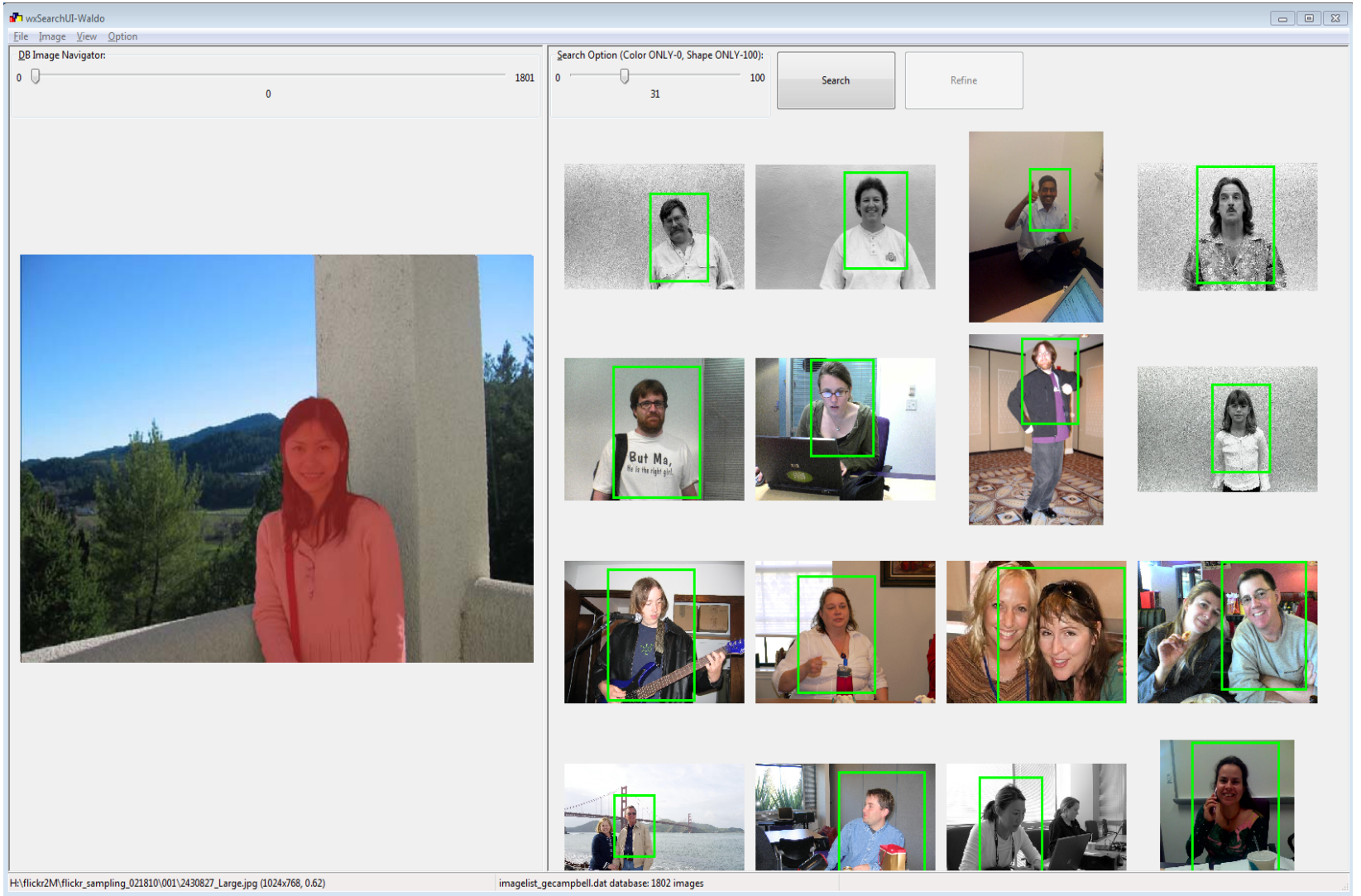
Facial Attribute Recognition





Data-Driven Object Segmentation

[J. Yang et al. CVPR 2014]



Data-Driven Automatic Cropping

[A. Samii et al. CGF 2015]

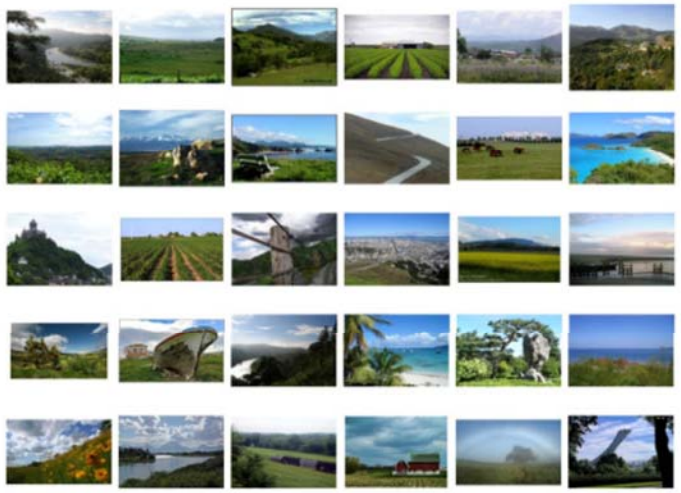


Image Similarity Search

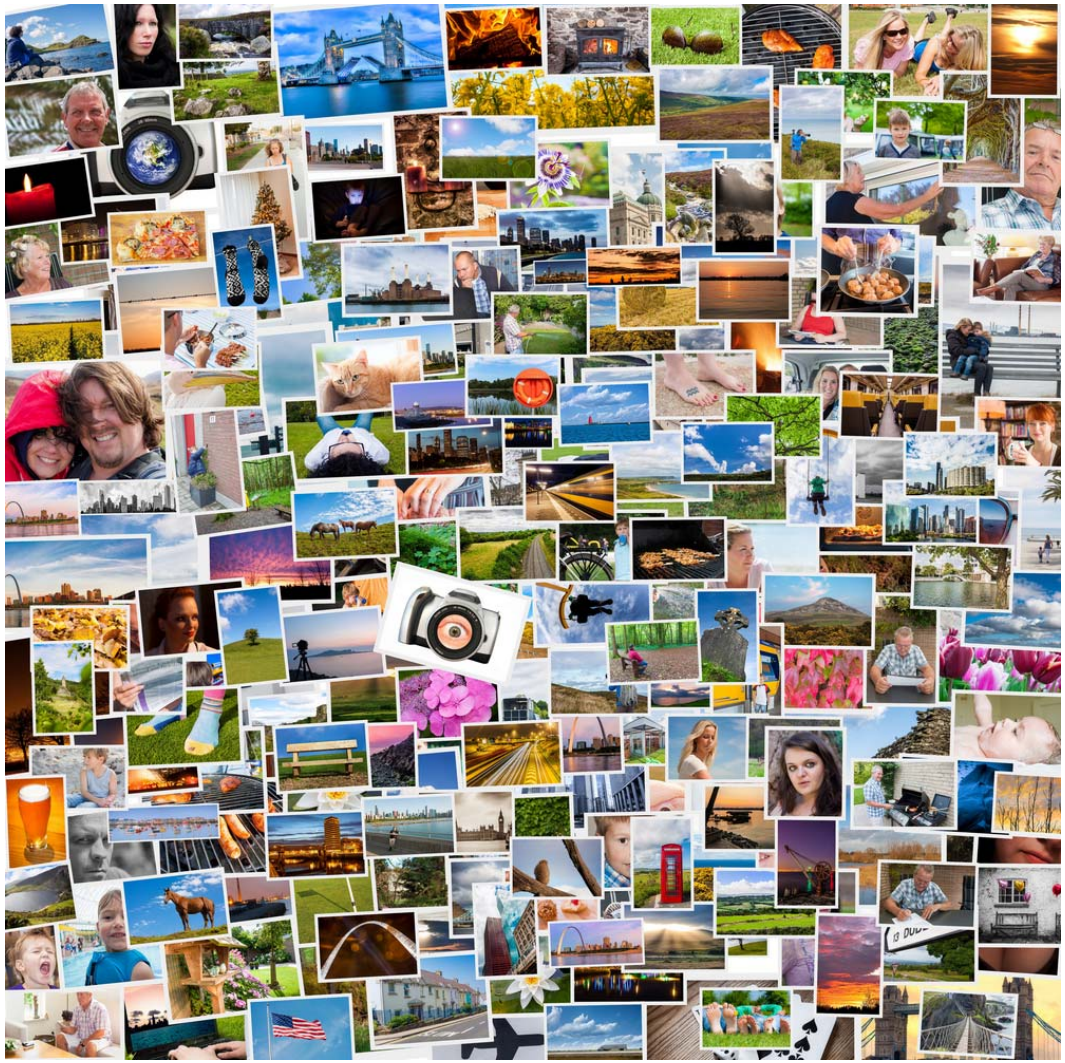
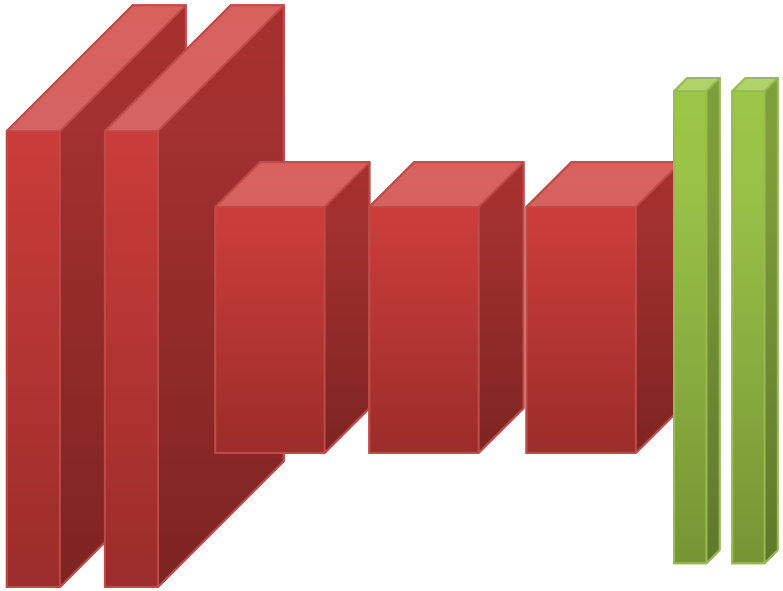


Image Similarity Search

Deep Convolutional Neural Network



Distance Encoded OPQ

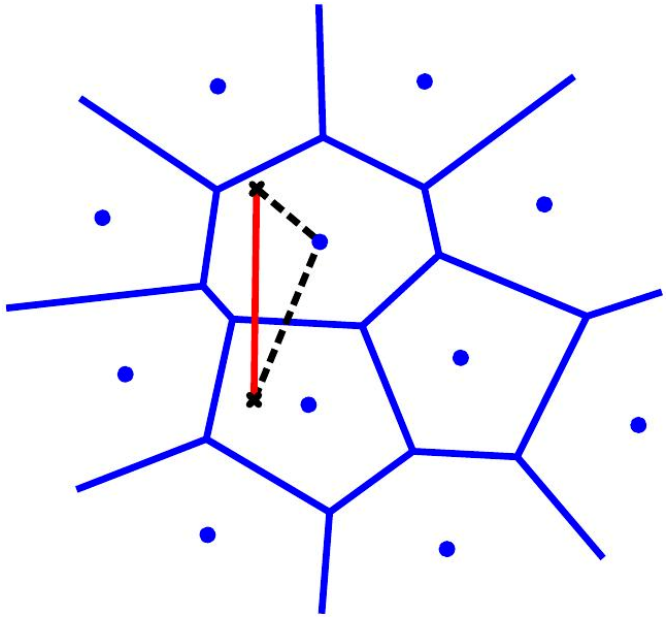
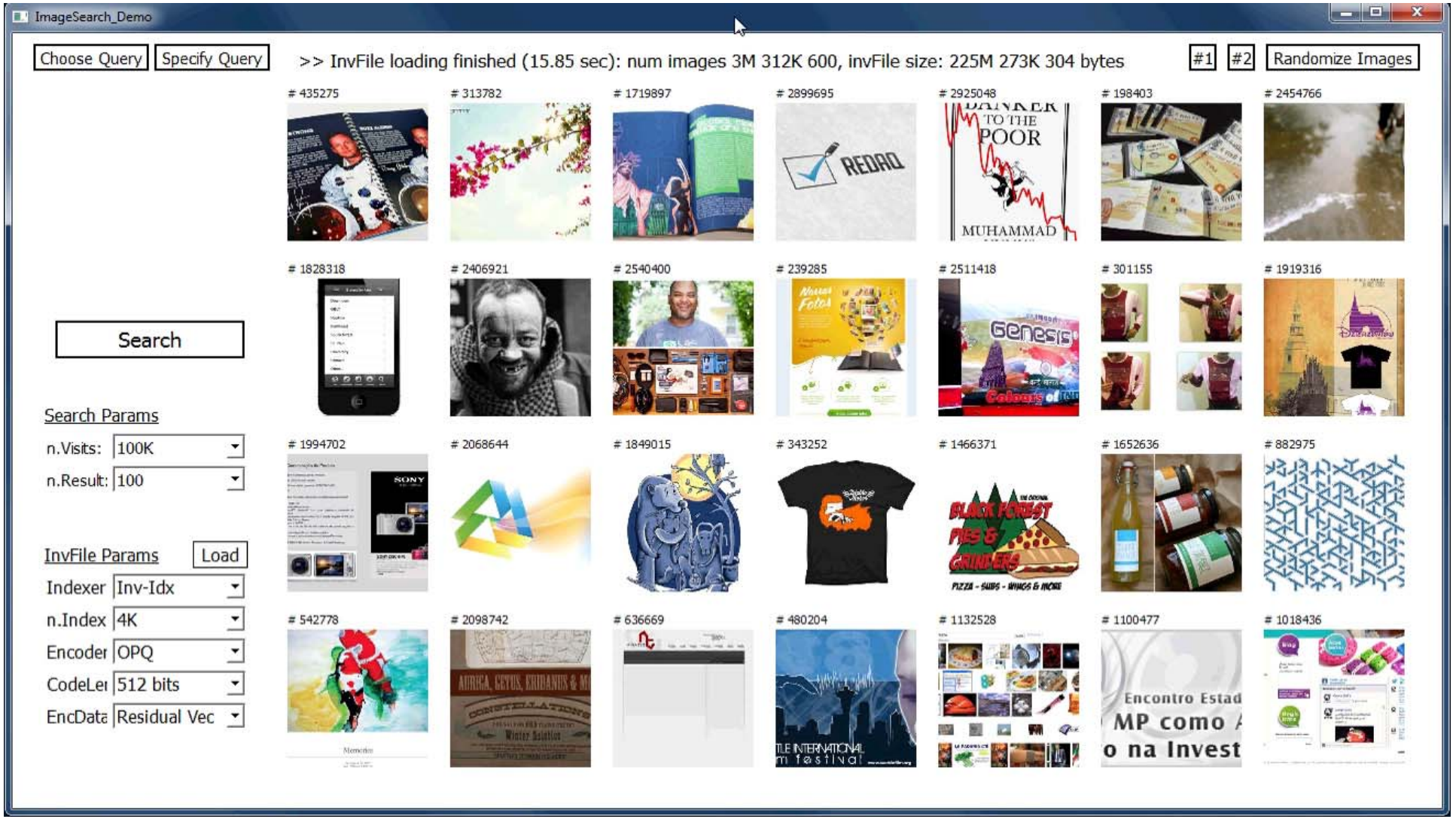


Image Similarity Search

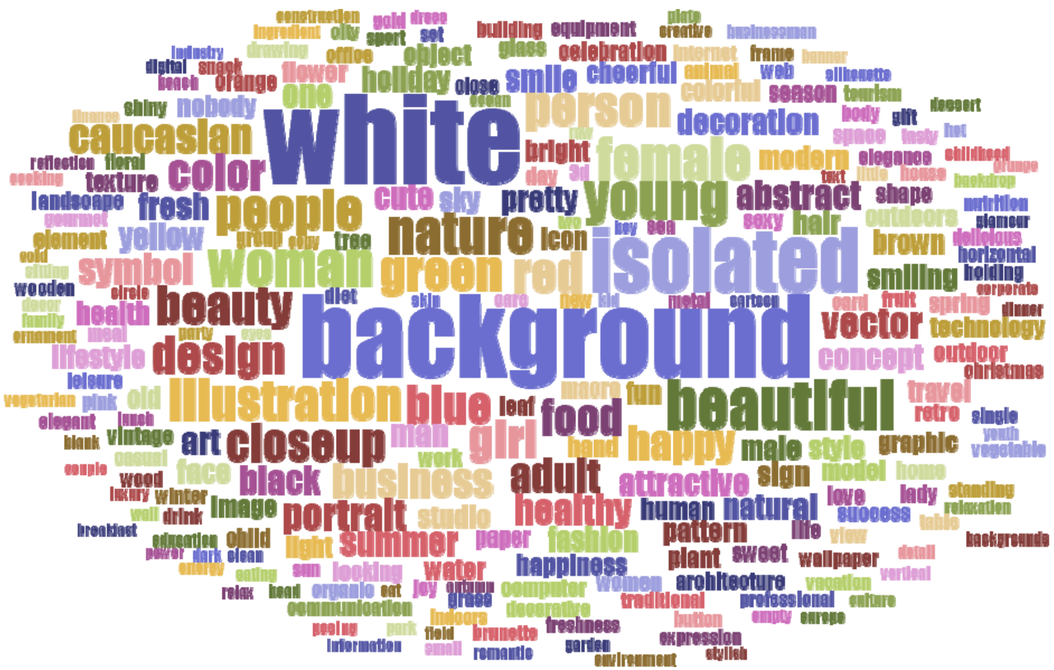


Automatic Image Tagging

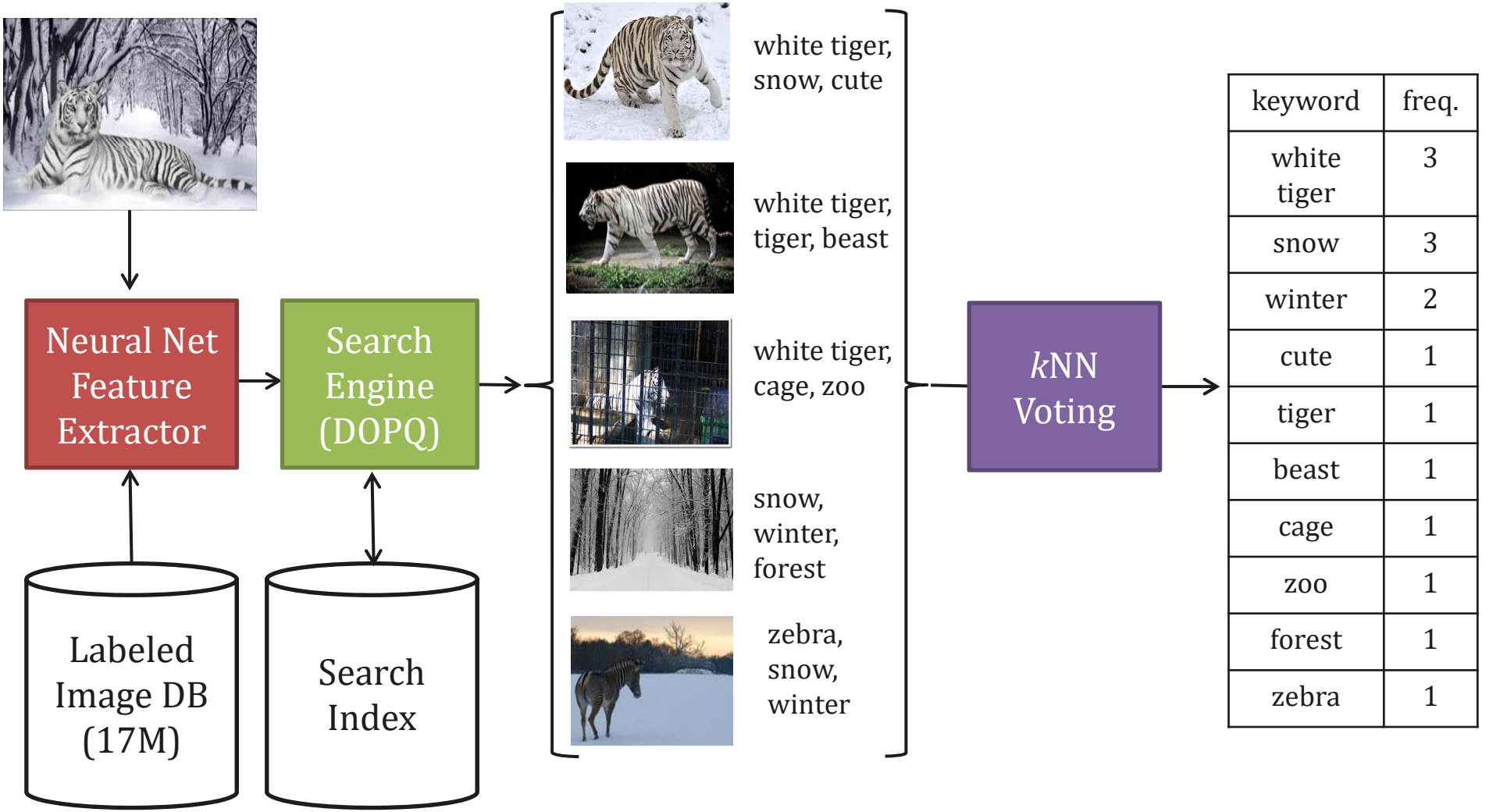
Dataset

- 17 million Adobe stock images with tags

18k-100k tag vocabulary



Deep-kNN Tagging System



keyword	freq.
white tiger	3
snow	3
winter	2
cute	1
tiger	1
beast	1
cage	1
zoo	1
forest	1
zebra	1

Discriminative Feature Learning

- Tag set similarities can reflect the visual similarities

cute, fluffy, domestic, one, playful, curious, portrait , funny, paw, young, spots, black, pets, puppy, adora ble, pretty, sitting, charming, fur, hair, lovely, anima ls, look, heartwarming, delectable, cuddly, veterina rian, small, mammal, soft, exquisite, ears, black pu ppy

cute, wreath, domestic, one, playful, bright, young, spots, black, pretty, orange, lovely, hawaiian, sittin g, motley, charming, fur, flowers, animals, look, hea rtwarming, cuddly, veterinarian, small, mammal, s oft, exquisite, ears, fluffy, hair, portrait, curious, fun ny, paw, artificial, pets, puppy, adorable, delectable , small black, black puppy

Tag Context Space

- We define pseudo classes from tag information

Image-Tags

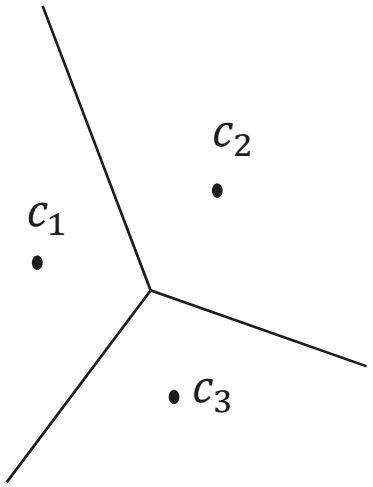
M dim sparse Bag of Tags

$$I_1 = \{t_1^1, t_2^1, \dots\} \quad x_1 = [0,1,1,0, \dots]$$
$$I_2 = \{t_1^2, t_2^2, \dots\} \quad x_2 = [1,0,0,0, \dots]$$
$$\vdots \qquad\qquad \vdots$$
$$I_n = \{t_1^n, t_2^n, \dots\} \quad x_n = [0,0,1,1, \dots]$$

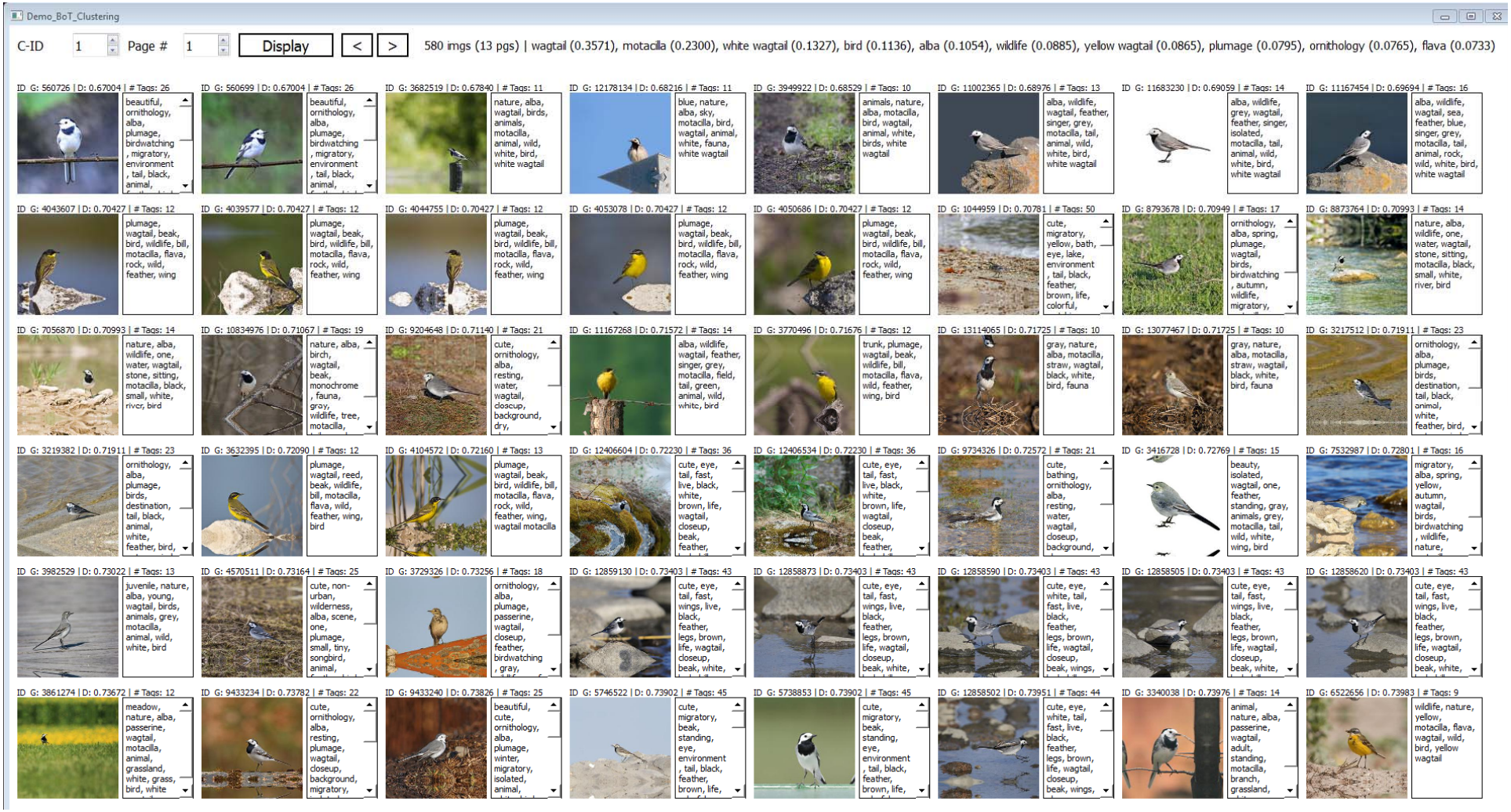
tf/idf
L_2 normalize

clustering

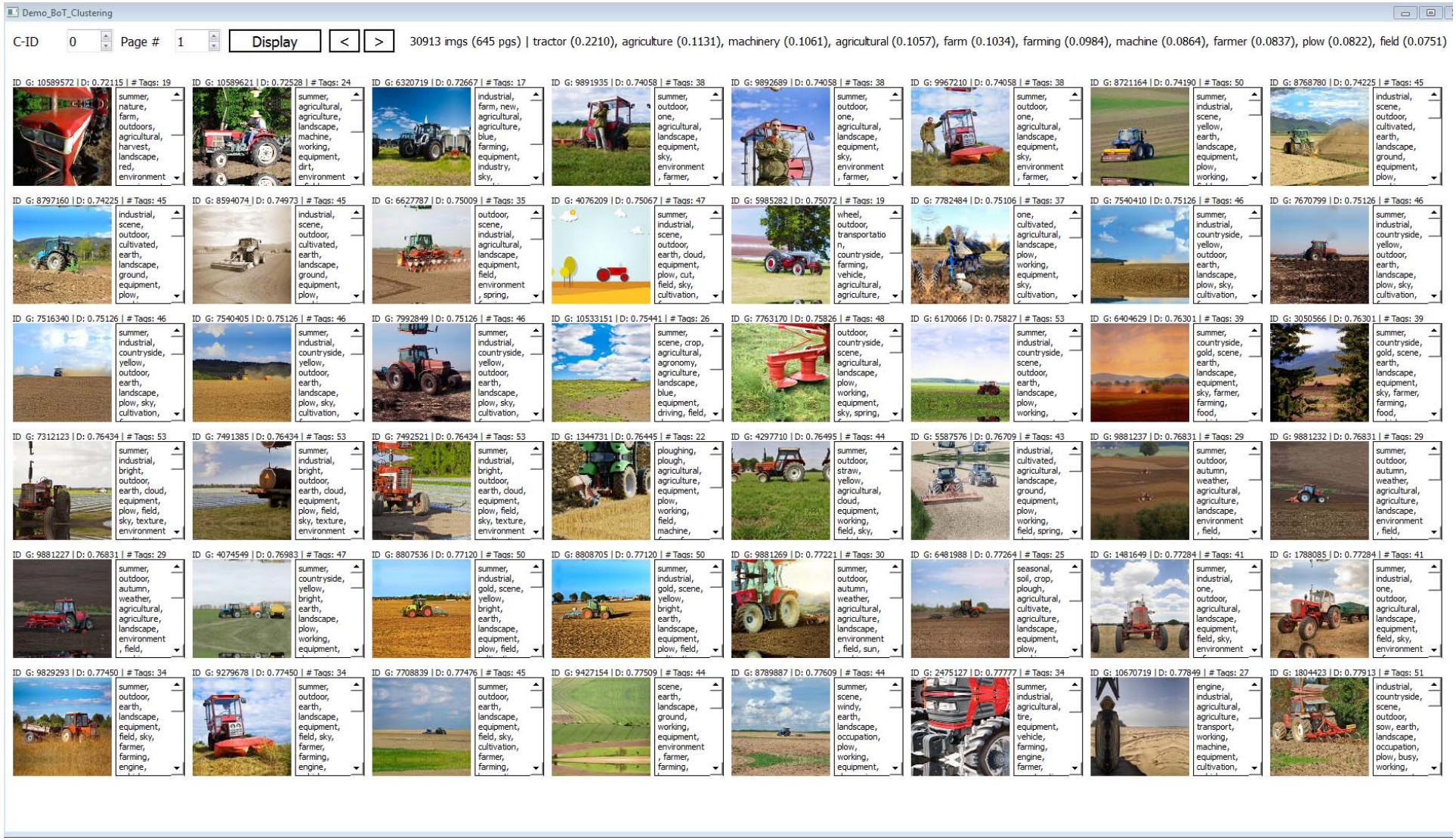
k pseudo classes



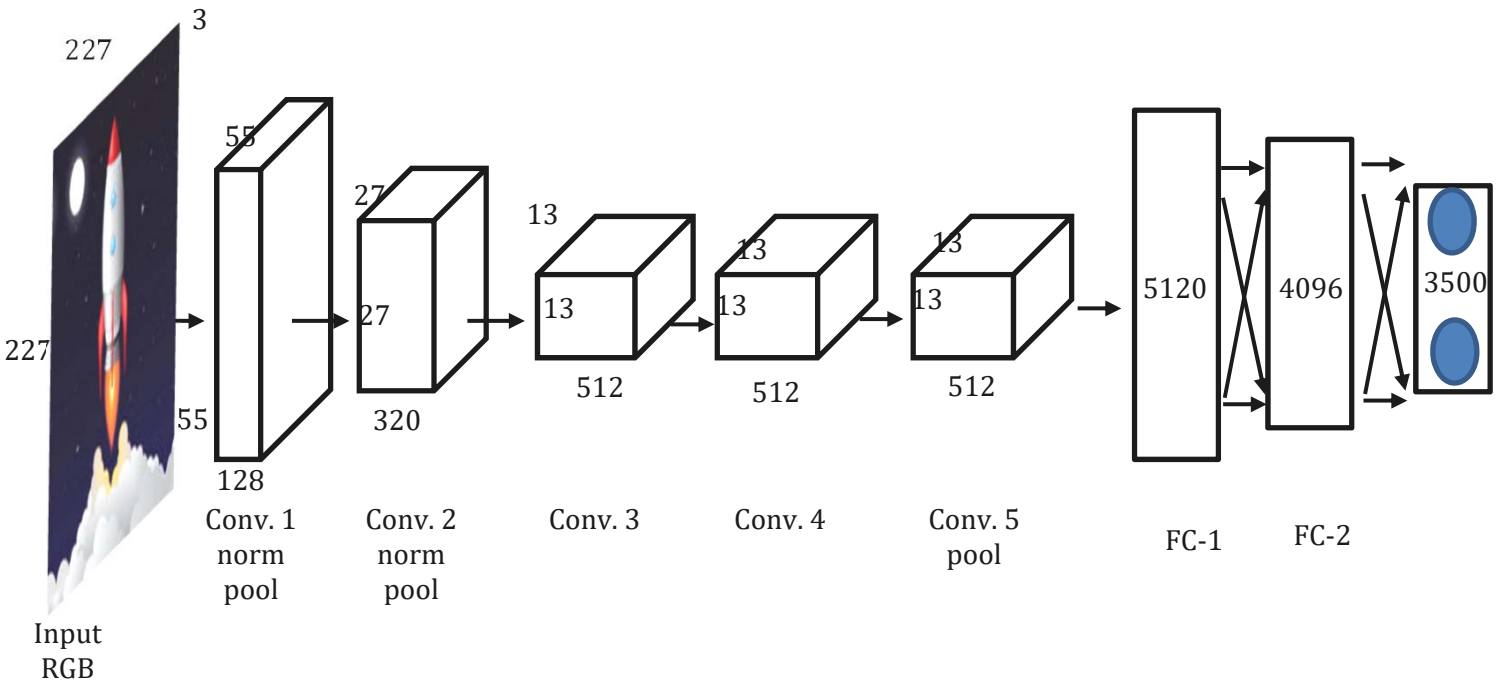
Sudo-Classes



Sudo-Classes



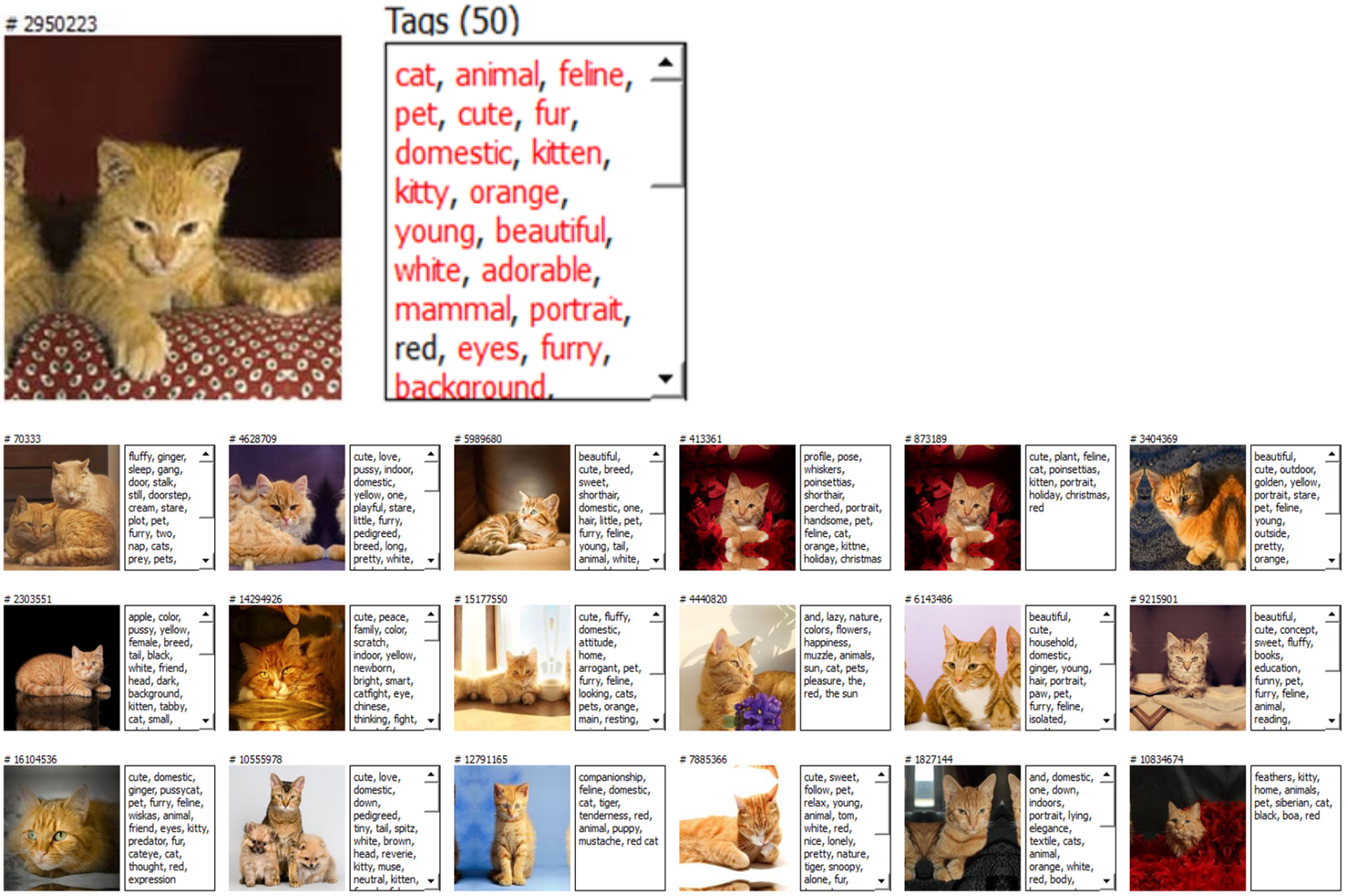
Deep Feature Learning



Tag Clusters

	100 Tags	
	P@100	R@100
ImageNet Feat (1000)	0.196	0.552
Deep Tagging Feat (1000)	0.204	0.573
Deep Tagging Feat (3500)	**0.220**	**0.616**

Automatic Image Tagging



Improving Tag Prediction

fish ?

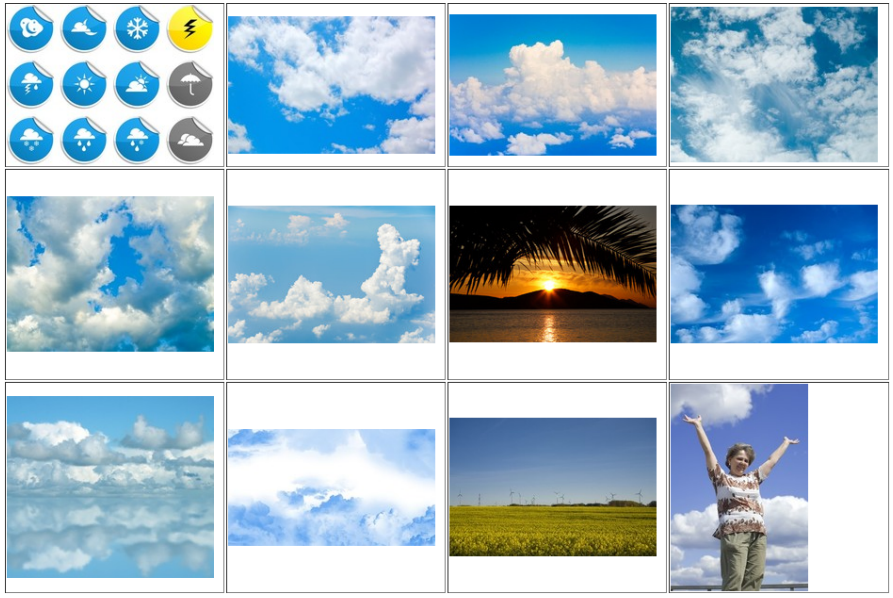
Highly Similar User Tags

fish
cook
cooking
man
food
kitchen
homework
....

Search Results

cloudy

KNN

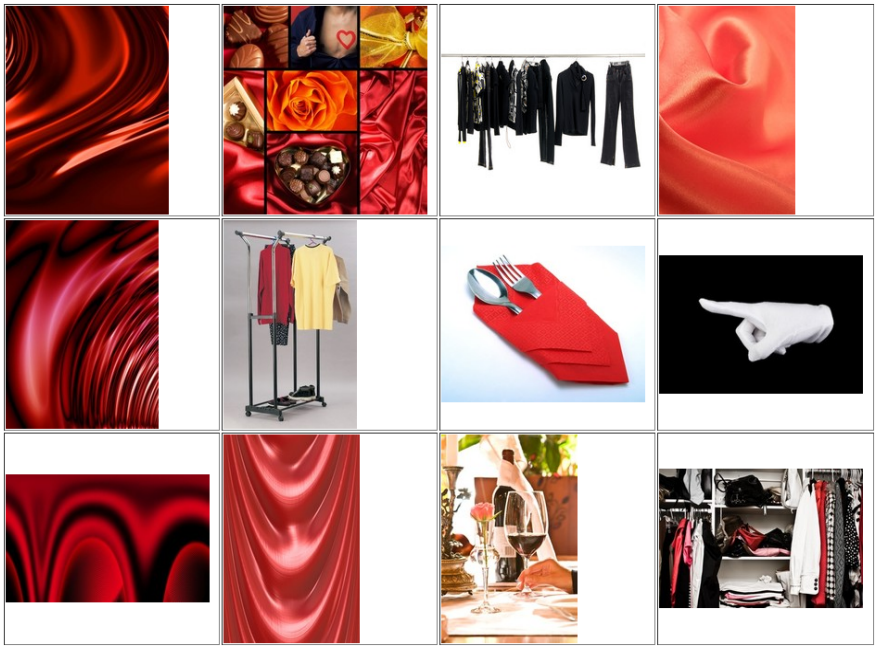


Div-KNN

Search Results

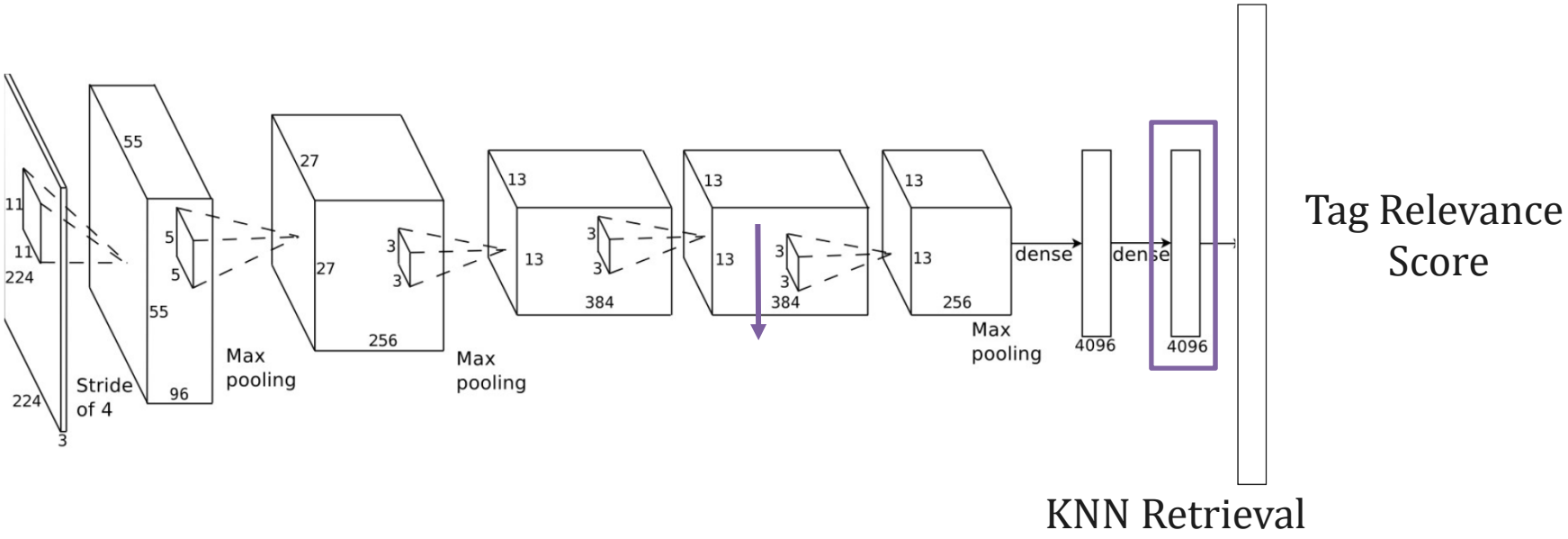
cloth

KNN



Div-KNN

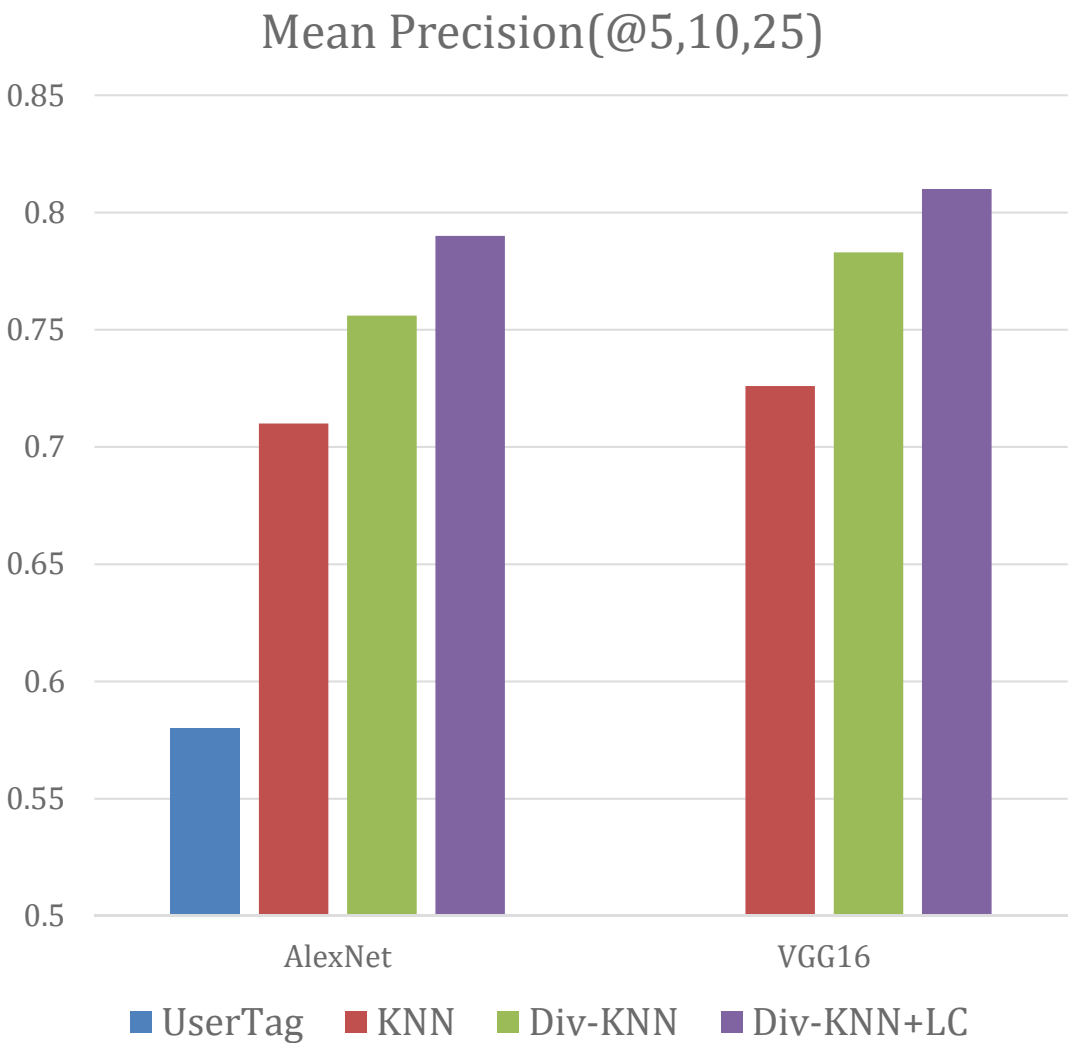
KNN + Linear Classifier



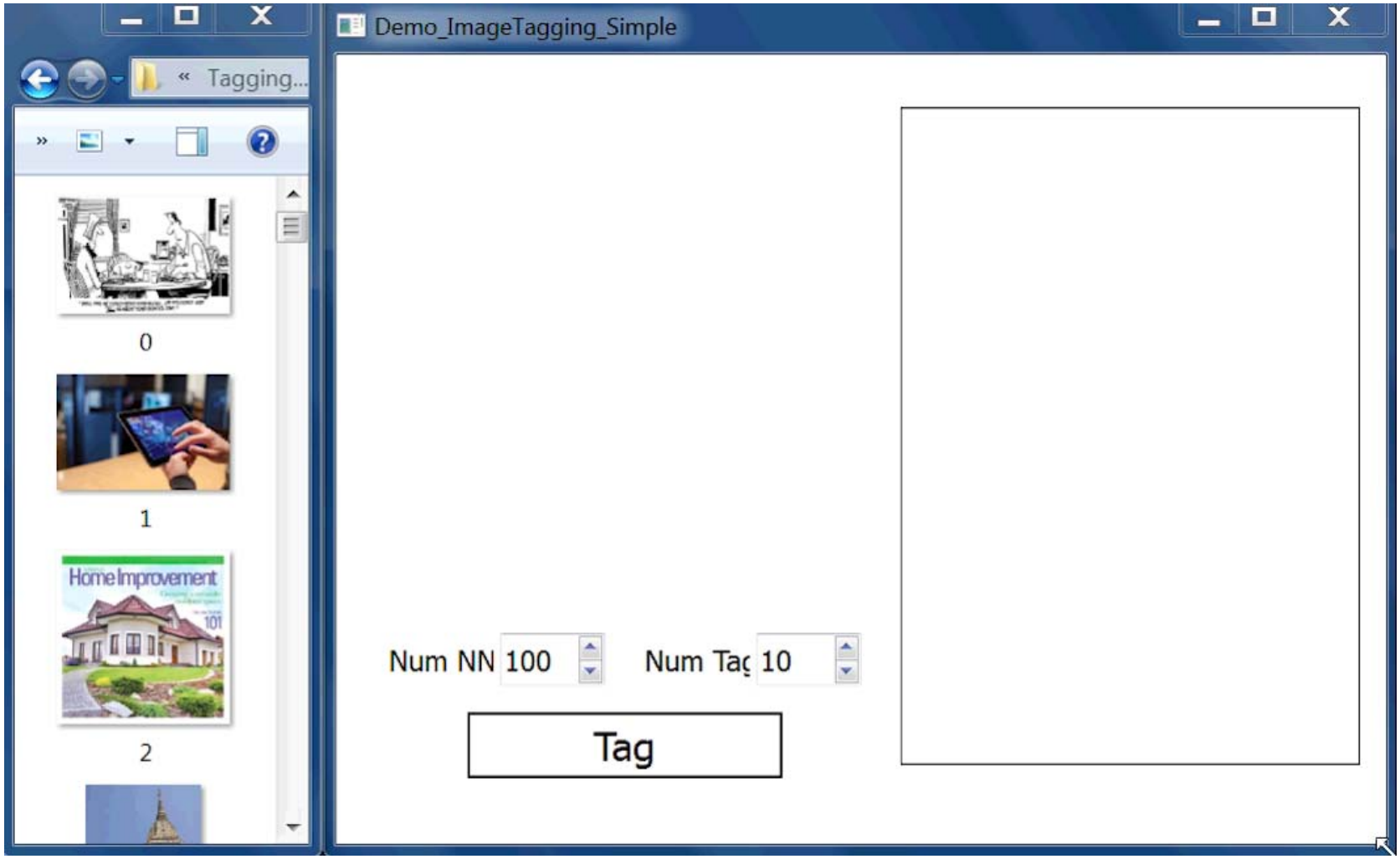
Linearly combine the scores from KNN-retrieval and Logistic classifier

Evaluation

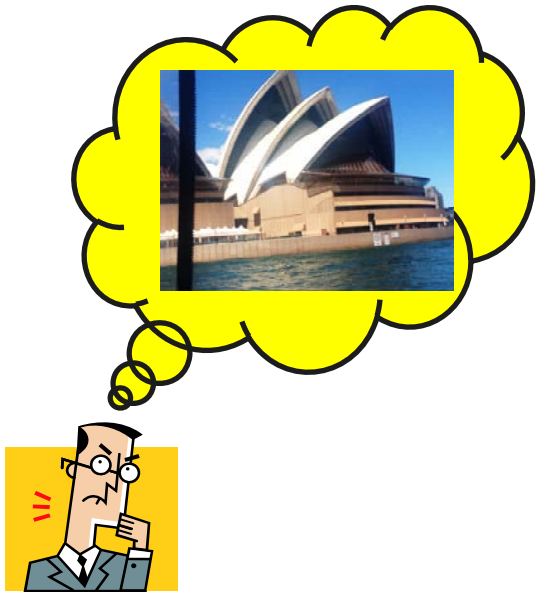
- Evaluation set
  - 82 tags: 30~50 images/tag
  - Include hard negatives
  - Top N Precision

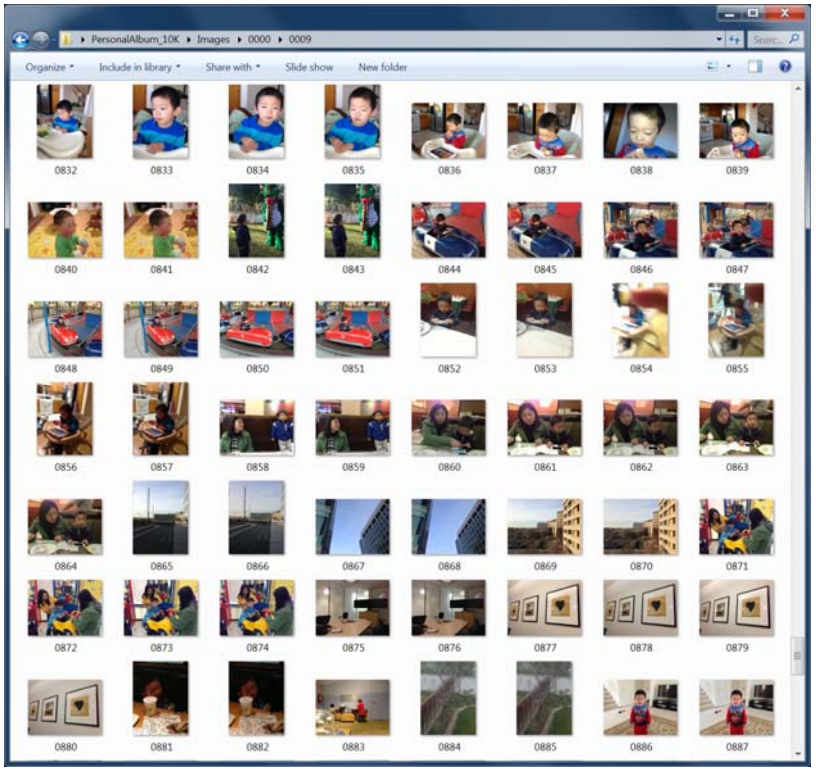


Demo


Tagging...
0
1
Home Improvement
2
Demo_ImageTagging_Simple
Num NN 100
Num Tag 10
Tag

Free-Text Image Search





Free-Text Image Search

- Internet Image Search → Crawl Examples

- Multi-Query Visual Search → Retrieve Images

Demo

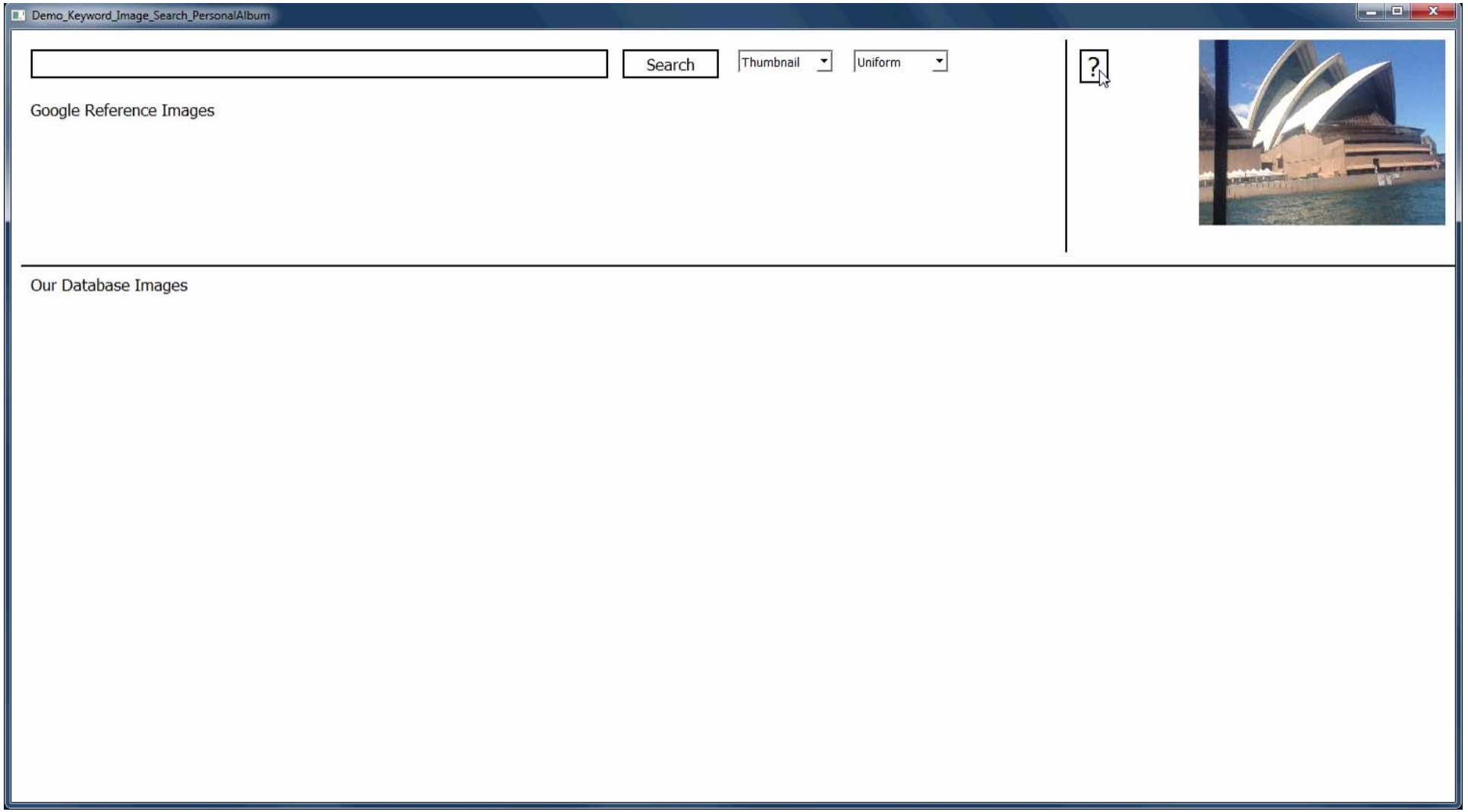

Demo_Keyword_Image_Search_PersonalAlbum
Search
Thumbnail
Uniform
Google Reference Images
Our Database Images

Image Recommendation

[C. Fang et al. CVPR 2015]

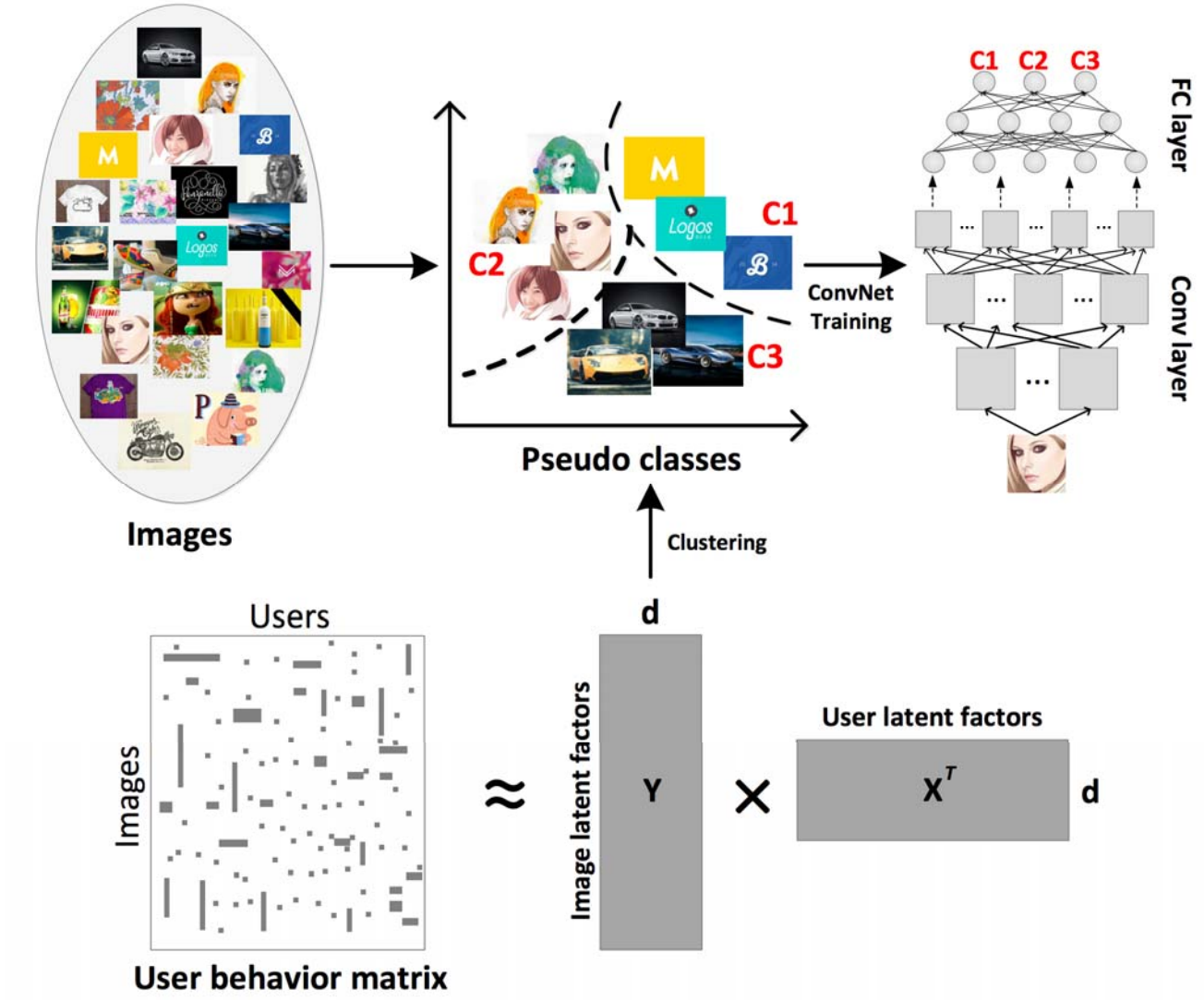


Image Recommendation

[C. Fang et al. CVPR 2015]