CS688/WST665: Web-Scale Image Retrieval

Recent Image Retrieval Techniques

Sung-Eui Yoon

(윤성의)

Course URL:
http://sglab.kaist.ac.kr/~sungeui/IR

Today

- Go over some of recent image retrieval techniques

Video Google: A Text Retrieval Approach to Object Matching in Videos

Josef Sivic and Andrew Zisserman

Robotics Research Group, Department of Engineering Science

University of Oxford, United Kingdom

ICCV 03

Citation: over 1300 at 2011

Motivations

- Retrieve key frames and shots of a video containing a particular object
- Investigate whether a text retrieval approach can be successful for object recognition

Viewpoint Invariant Description

- Find viewpoint covariant regions
  - Produce elliptical affine invariant regions (e.g., Shape Adapted (SA) and Maximally Stable(MS))
  - SA regions centered on corner like features
  - MS regions correspond to high contrast with respect to their surroundings (dark window, gray wall…)
- Compute a SIFT descriptor for each region

MSER(Maximally Stable Extremal Regions)

- Affinely-invariant stable regions in the image
  - can be used to localize regions around keypoints
  - We will use only SIFT descriptors that are inside of MSER regions

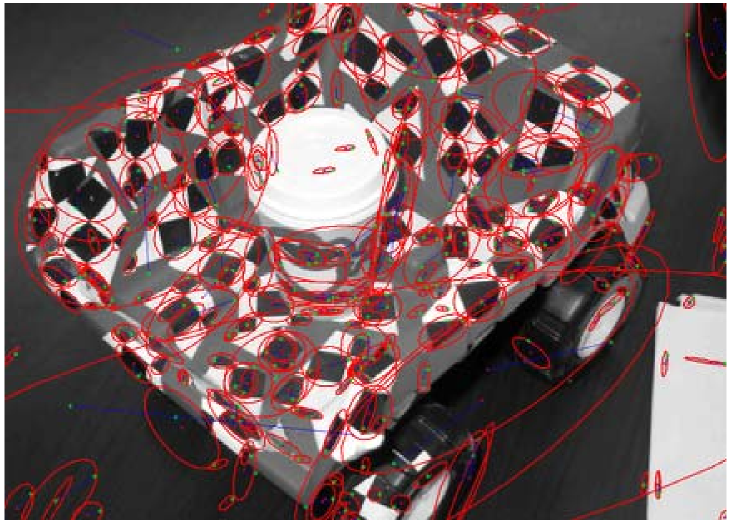


Figure 1: Top row: Two frames showing the same scene from very different camera viewpoints (from the film 'Run Lola Run'). Middle row: frames with detected affine invariant regions superimposed. 'Maximally Stable' (MS) regions are in yellow. 'Shape Adapted' (SA) regions are in cyan. Bottom row: Final matched

Visual Vocabulary

- **Quantize descriptor vectors into clusters, which are visual ‘word’ for text retrieval**
  - **Performed with K-means clustering**

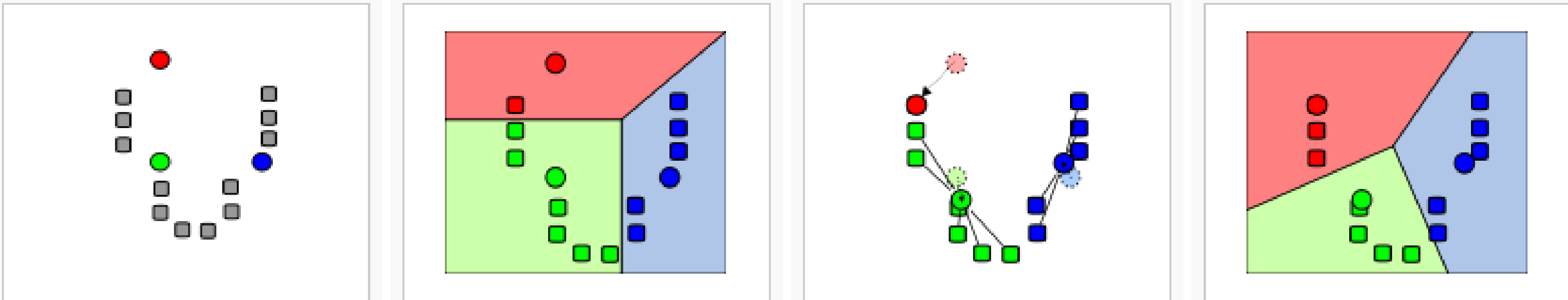
- **Produce about 6K and 10K clusters for Shape adapted and Maximally Stable regions respectively**
  - **Chosen empirically to maximize retrieval results**

K-Means Clustering

- **Minimize the within-cluster sum of squares (WCSS)**

$$\underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in S_i} \left\| \mathbf{x}_j - \boldsymbol{\mu}_i \right\|^2$$

Demonstration of the standard algorithm



1) *k* initial "means" (in this case *k*=3) are randomly selected from the data set (shown in color).

2) *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.

3) The centroid of each of the *k* clusters becomes the new means.

4) Steps 2 and 3 are repeated until convergence has been reached.

Distance Function

- **Use Mahalanobis distance as the distance function for clustering:**

$$d(\vec{x},\vec{y}) = \sqrt{(\vec{x}-\vec{y})^T S^{-1} (\vec{x}-\vec{y})}.$$

, where S is covariance matrix

  - **If S is the identify matrix, it reduces to Euclidean distance**
  - **Decorrelate components of SIFT**

- **Instead, Euclidean distance may be used**

Visual Indexing

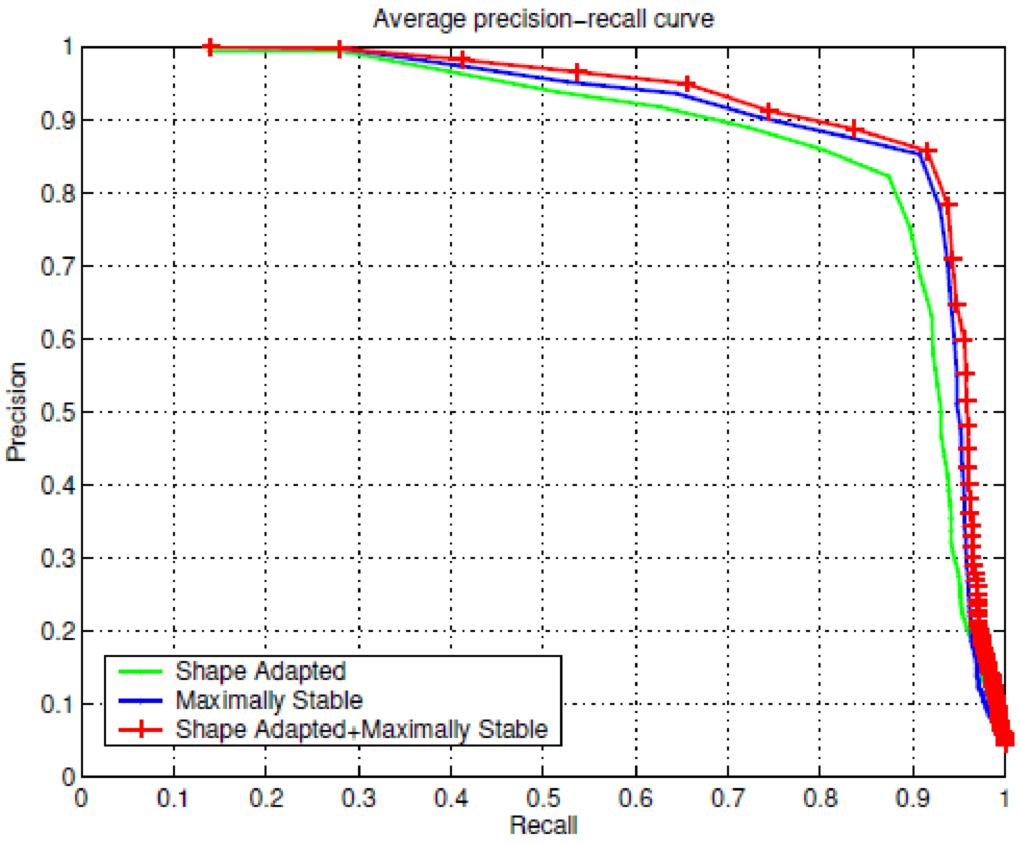
- Each document is represented by k-vector $(t_1, ..., t_i, ..., t_k)^\top$
- Weighting by tf-idf
  - term frequency * log (inverse document frequency)

$$t_i = \frac{n_{id}}{n_d} \log \frac{N}{n_i}$$

  - n_{id} : # of occurrences of word i in document d
  - n_d : total # of words in the document d
  - n_i : # of occurrences of term i in the whole database
  - N: # of documents in the whole database
- At the retrieval stage documents are ranked by their normalized scalar product between query vector V_q and V_d in database

Video Google [Sivic et al. CVPR 2003]

- **mAP: mean average precision**



Video Google [Sivic et al. CVPR 2003]

- **Performance highly depended on number of k(visual words) : not scalable**

KAIST

Scalable Recognition with a Vocabulary Tree

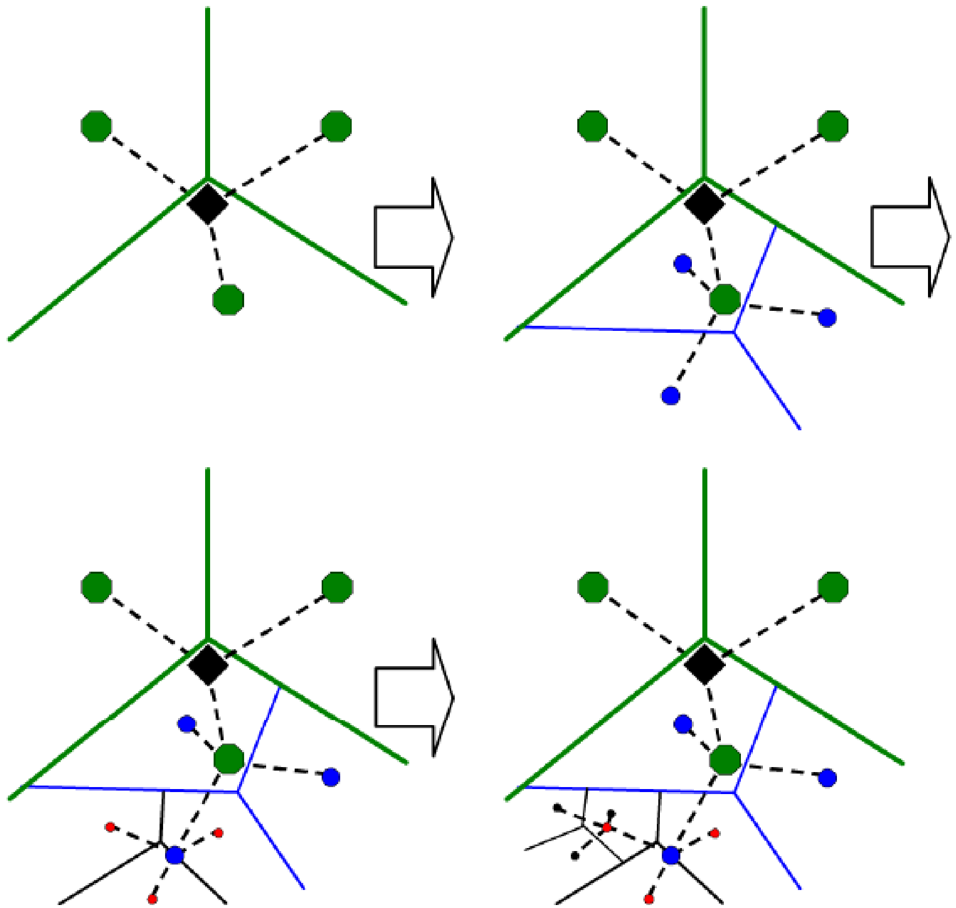
David Niter et al.

CVPR 2006

Citation: over 1000 at 2011

Vocabulary Tree [Nister et al. CVPR 06]

- **Hierarchical k-means clustering**



Vocabulary tree with branch factor 10

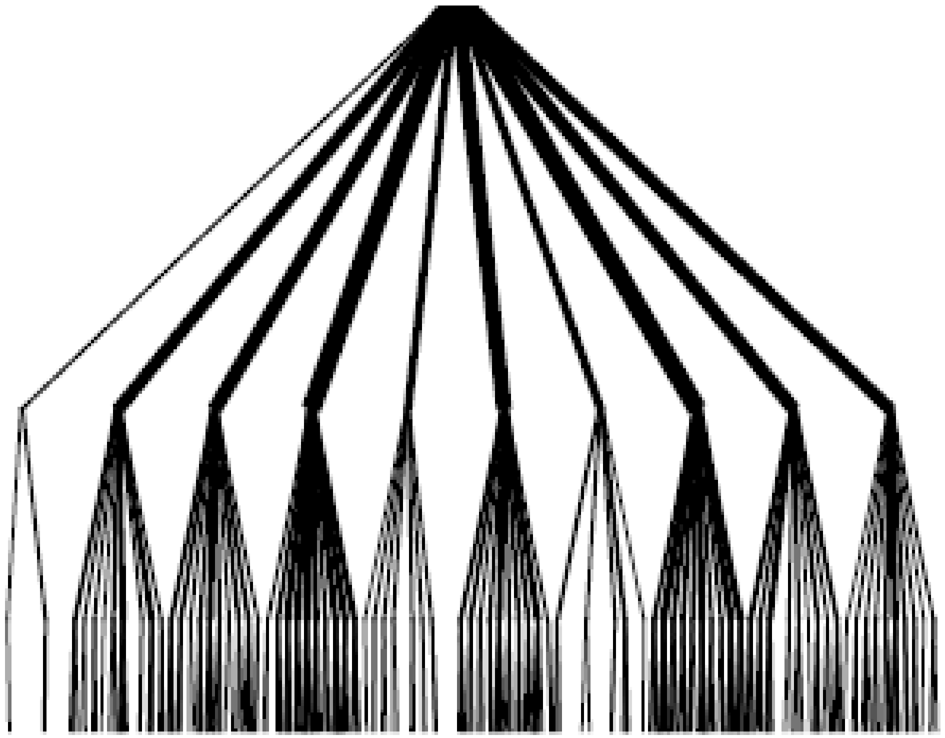


Figure 3. Three levels of a vocabulary tree with branch factor 10 populated to represent an image with 400 features.

Inverted File

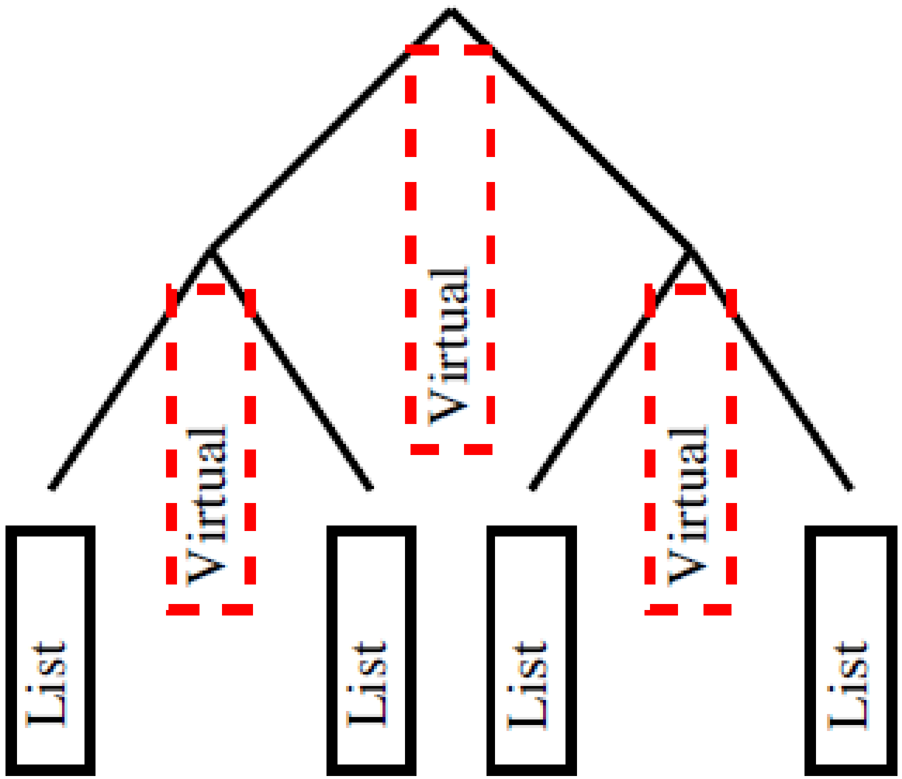
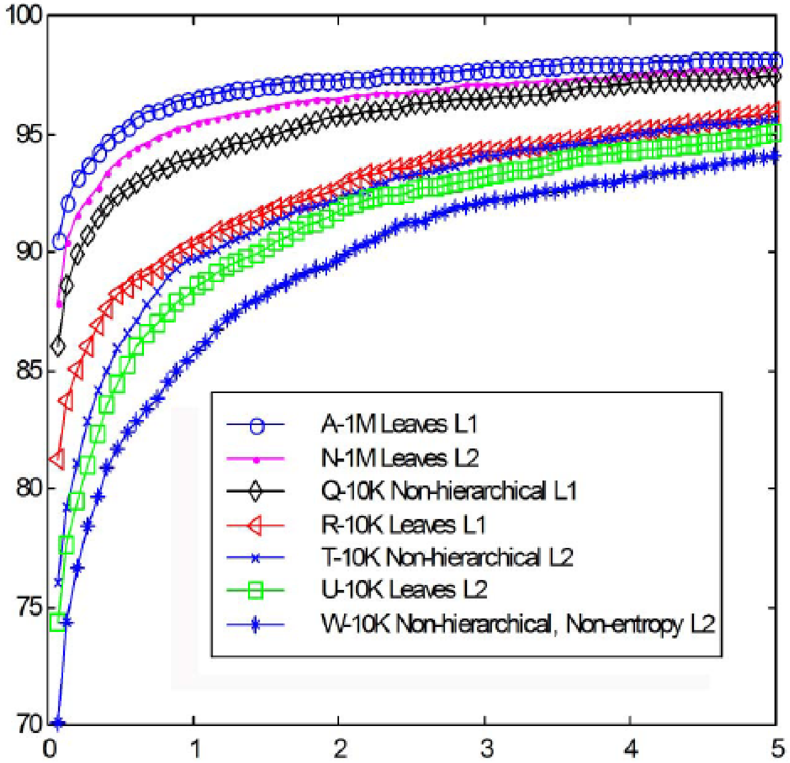


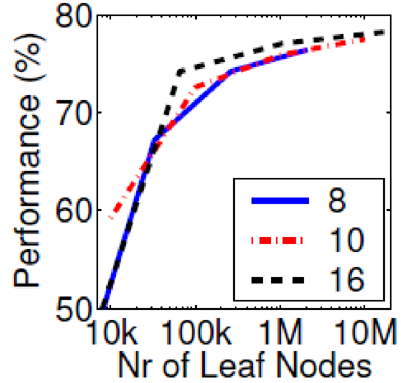
Figure 4. The database structure shown with two levels and a branch factor of two. The leaf nodes have explicit inverted files and the inner nodes have virtual inverted files that are computed as the concatenation of the inverted files of the leaf nodes.

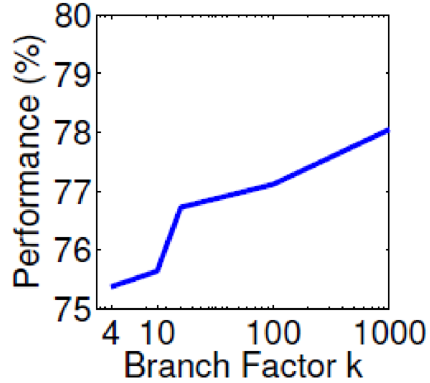
Retrieval Algorithm

- Compute a histogram of visual words with SIFTs
- Identify images that contain words of the input query image
  - Can be done with the inverted file
- Sort images based on a similarity function

Vocabulary Tree [Nister et al. CVPR 06]







- On 8GB RAM machine(40000 images)queries took 1s, database creation took 2.5 days

Vocabulary Tree

- **Benefits:**
  - **Allow faster image retrieval (and pre-computation)**
  - **Scales efficiently to a large number of images**
- **Problems:**
  - **Too much memory requirement**
  - **Quantization effects**

Object retrieval with large vocabularies and fast spatial matching

Philbin et al.

CVPR 2007

Citation: over 350 at 2011

Approximating K-means

- Use a forest of 8 randomized k-d trees
  - Randomize splitting dimension among a set of the dimensions with highest variance
  - Randomly choose a point close to the median for split value
  - Helps to mitigate quantization effects
- Each tree is descending to leaf, distance from boundaries are recorded in a prior queue
  - Similar to best-bin-first search

KAIST

Approximate K-means

- Algorithmic complexity of a single k-means iteration
  - Reduces from O(NK) to O(NlogK), where N is the # of features
  - Achieved by multiple random kd-trees
- Find images with kd-trees too

- But using approximate K-means, performance is superior!
  - Due to reduction of quantization effect)

Spatial Re-Ranking with RANSAC

- **Generate hypotheses with pairs of corresponding features**
  - **Assume a restricted transformation, since many images on the web are captured in particular ways (axis-aligned ways)**
- **Evaluate other pairs and measure errors**
- **Re-ranking images by scoring the # of inliers**

Transformation	dof	Matrix
translation + isotropic scale	3	$\begin{bmatrix} a & 0 & t_x \\ 0 & a & t_y \end{bmatrix}$
translation + anisotropic scale	4	$\begin{bmatrix} a & 0 & t_x \\ 0 & b & t_y \end{bmatrix}$
translation + vertical shear	5	$\begin{bmatrix} a & 0 & t_x \\ b & c & t_y \end{bmatrix}$

(a)

Method / Rerank N	100	200	400	800
i 3dof	0.468	0.492	0.522	0.556
ii 4dof	0.465	0.490	0.521	0.555
iii 5dof	0.467	0.491	0.526	0.560

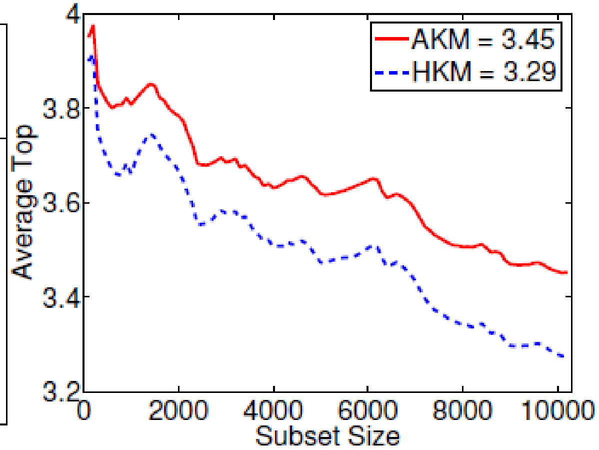
(b)

Method / Rerank N	100	200	400	800
i 3dof	0.644	0.650	0.652	0.655
ii 4dof	0.646	0.656	0.659	0.661
iii 5dof	0.648	0.657	0.660	0.664

Results

Clustering parameters		mAP	
# of descr.	Voc. size	k-means	AKM
800K	10K	0.355	0.358
1M	20K	0.384	0.385
5M	50K	0.464	0.453
16.7M	1M		0.618

Method	Scoring Levels	Average Top
HKM	1	3.16
HKM	2	3.07
HKM	3	3.29
HKM	4	3.29
AKM		**3.45**

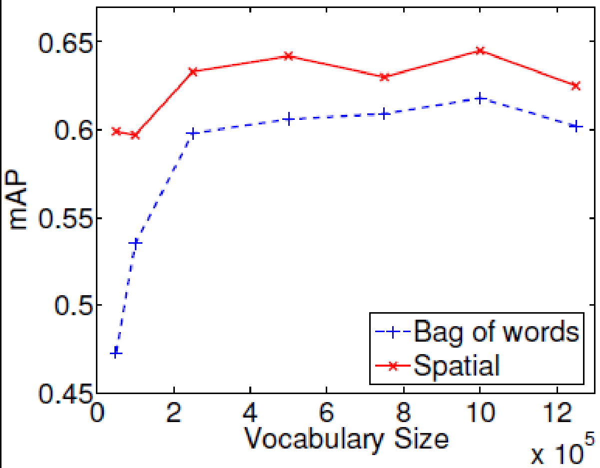


KAIST

Results

Method	Dataset	mAP	
		Bag-of-words	Spatial
(a) HKM-1	5K	0.439	0.469
(b) HKM-2	5K	0.418	
(c) HKM-3	5K	0.372	
(d) HKM-4	5K	0.353	
(e) AKM	5K	0.618	0.647
(f) AKM	5K+100K	0.490	0.541
(g) AKM	5K+100K+1M	0.393	0.465

Vocab Size	Bag of words	Spatial
50K	0.473	0.599
100K	0.535	0.597
250K	0.598	0.633
500K	0.606	0.642
750K	0.609	0.630
1M	**0.618**	**0.645**
1.25M	0.602	0.625



Total Recall: Automatic Query Expansions with a Generative Feature Model for Object Retrieval

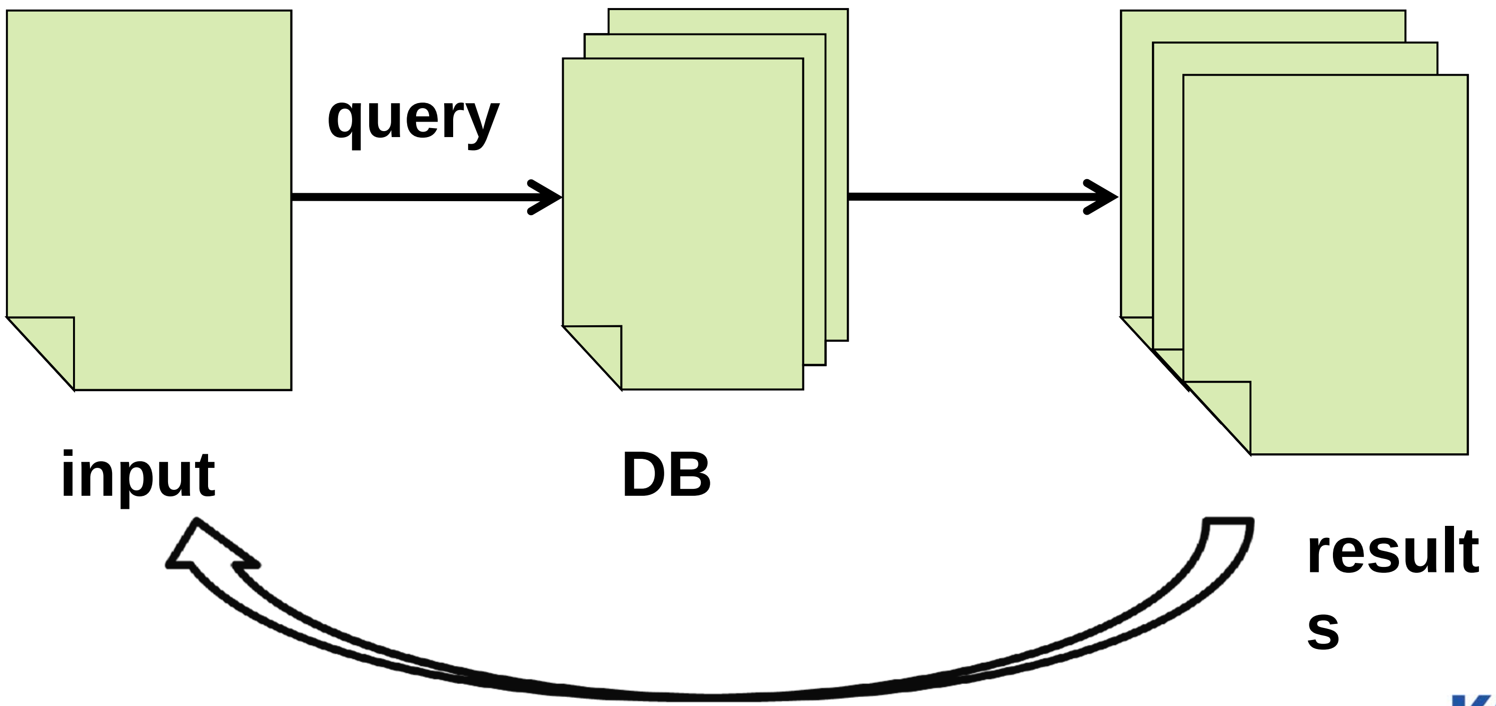
Chum et al.

ICCV 2007

Citation: over 150 at 2011

Query Expansion

- Improve recall with re-querying combination of the original query and result with spatial verification



Query Expansion

- Spatial verification
  - Similar with the technique used in [Philbin et al. 07]; Uses a RANSAC-like algorithm
  - Identify a set of images that are very similar to the original query image

BoW interpreted Probabilistically

- Extracts a generative model of an object from the query region
- Compute a response set that are likely to have been generated from the model
- The generative model
  - Spatial configuration of visual words with a background clutter

Generative Models

- **Query expansion baseline**
  - **Average term frequency vectors from the top 5 queries without verification**
- **Transitive closure expansion**
  - **A priority queue of verified images is keyed by # of inliers**
  - **Take the top image and query it as a new query**
- **Average query expansion**
  - **A new query is constructed by averaging the top 50 verified results (di is the term frequency vector of ith verified image)**

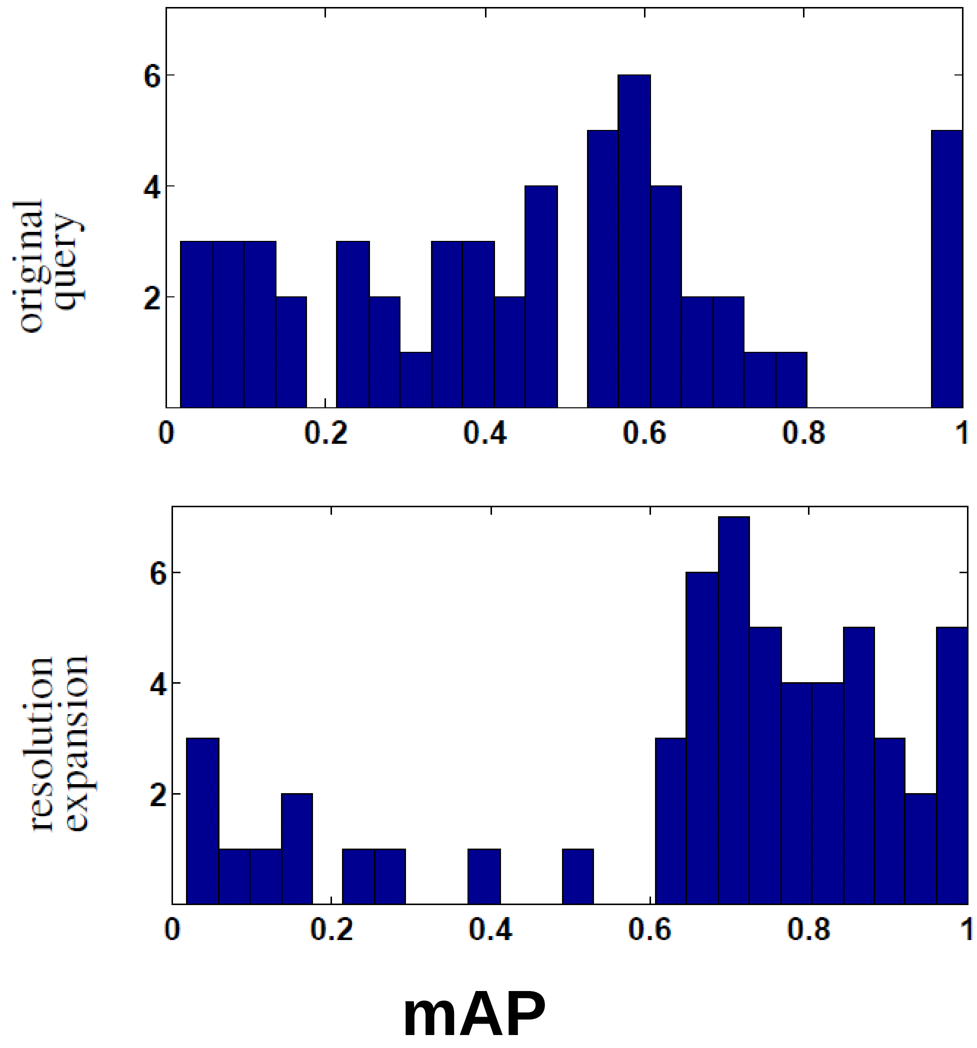
$$d_{\text{avg}} = \frac{1}{m+1}\left(d_0 + \sum_{i=1}^{m} d_i\right)$$

KAIST

Generative Models

- **Multiple image resolution expansion**
  - **Consider images with different resolutions; higher resolutions give more detailed information**
  - **Use a resolution band with (0, 4/5), (2/3, 3/2), and (5/4, infinity)**
  - **Use averaged queries for each resolution band**
  - **Show the best result**

Results



Results

Original query

Top 4 images

Expanded results that were not identified by the original query

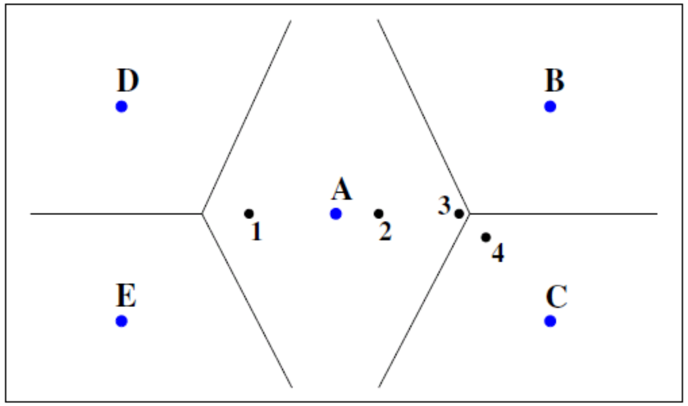
Lost in Quantization: Improving Particular Object Retrieval in Large Scale Image Databases

Philbin et al.

CVPR 2008

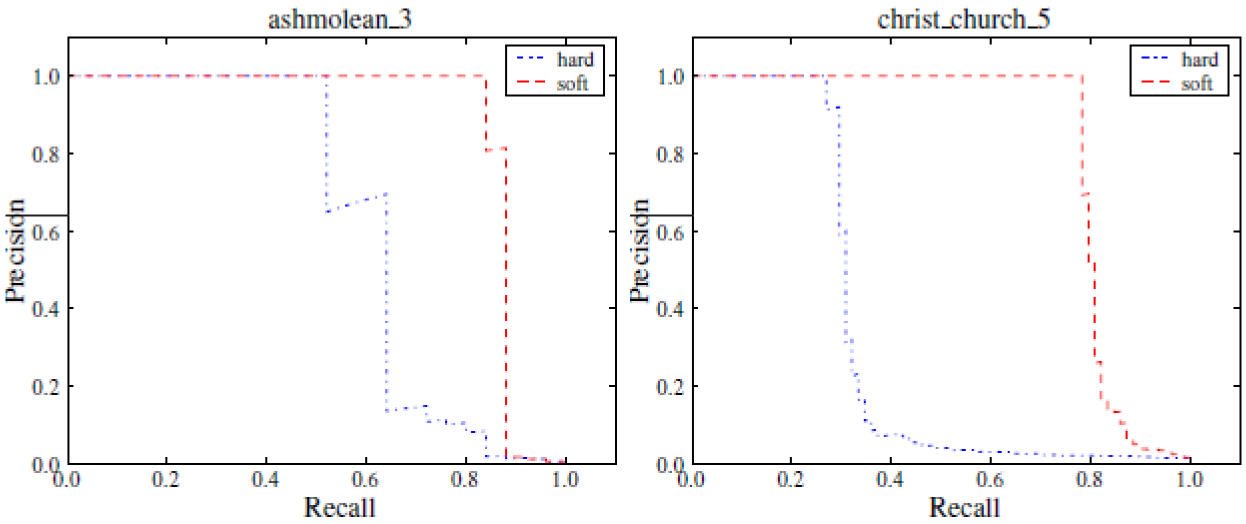
Citation: over 175 at 2011

Soft Quantization [Philbin et al. CVPR 08]



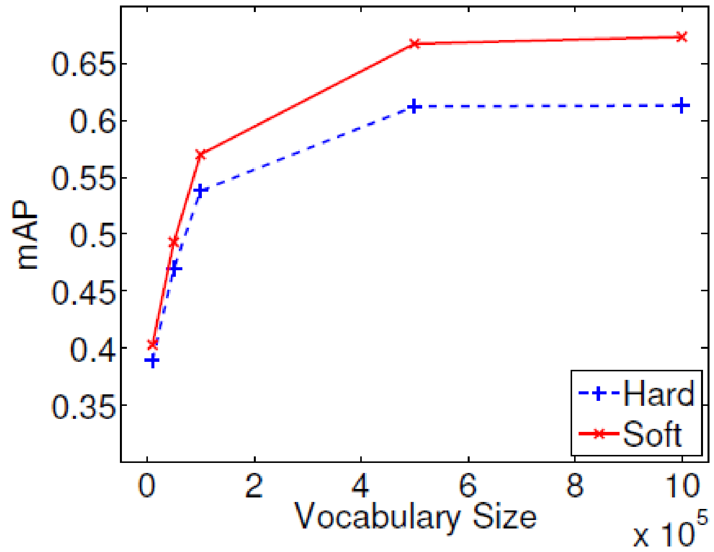
- 3 and 4 will be never matched in hard assignment
- No way of distinguishing 2 and 3 are closer than 1 and 2
- Soft assignment: use a weight vector
  - A weight to a cluster is assigned proportional to the distance between the descriptor and the center of the cluster

Results



Method	Training data Oxford	Paris
Fixed Quantization [18]	0.164	
HKM [14] (1 level)	0.422	0.401
HKM [14] (2 level)	0.410	0.340
Hard [15]	0.614	0.403
Soft	**0.673**	**0.494**

Effect of Vocabulary Size and Number of Images



- **For Oxford dataset with 1M vocabulary, hard assignment index costs 36MB and soft costs 108MB with compression**

City-Scale Location Recognition

Schindler et al.

CVPR 2007

Citation: over 135 at 2011

City-Scale Location Recognition

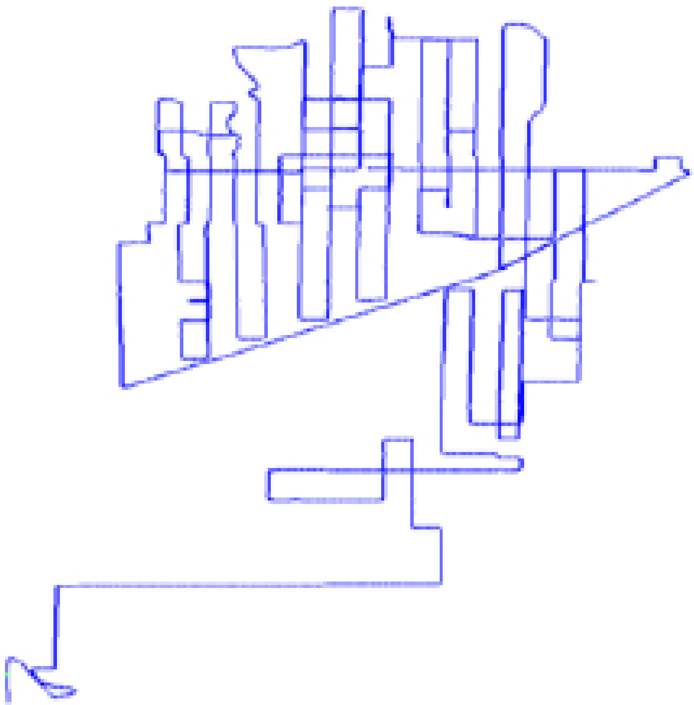


Figure 1. We perform location recognition on 20 km of urban streetside imagery, storing 100 million features in a vocabulary tree, the structure of which is determined by the features that are most informative about each location. Shown here is the path of our vehicle over 20 km of urban terrain.

Example Image Database

Figure 8. Example database image sequences from commercial (top), residential (middle), and green (bottom) areas of a city. The significant overlap between consecutive images allows us to determine which features are most informative about each location.

Challenges and Main Ideas

- Too many images
  - Storage-space and search –time problems

- Main approaches
  - Use a vocabulary tree to organize millions of feature descriptors
  - Choose more informative image sets for identifying locations, instead of organizing all the images

Informative Features

- Want to find features
  - Occur in all images of specific locations
  - But, rarely or never occur anywhere outside of that single location
- Can be captured formally in information gain
  - How much uncertainty is removed by additional knowledge

Information Gain

- **How much uncertainty is removed by additional knowledge**

$$H(X) = -\sum_x P(X = x) \log[P(X = x)]$$ **Entropy**

$$H(X|Y) = \sum_y P(Y = y) H(X|Y = y)$$ **Conditional entropy**

$$I(X|Y) = H(X) - H(X|Y)$$ **Information gain**

$$I(L_i|W_j) = H(L_i) - \boxed{H(L_i|W_j)}$$

Visual word

We want to minimize it

Binary value when we are at a particular location

Fewer Bits

Someone tells you that the probabilities are not equal

$P(X=A) = 1/2$	$P(X=B) = 1/4$	$P(X=C) = 1/8$	$P(X=D) = 1/8$

It's possible…

…to invent a coding for your transmission that only uses 1.75 bits on average per symbol. How?

Fewer Bits

Someone tells you that the probabilities are not equal

$P(X=A) = 1/2$	$P(X=B) = 1/4$	$P(X=C) = 1/8$	$P(X=D) = 1/8$

It's possible…

…to invent a coding for your transmission that only uses 1.75 bits on average per symbol. How?

A	0
B	10
C	110
D	111

(This is just one of several ways)

General Case

Suppose X can have one of *m* values… $V_1, V_2, \ldots V_m$

$P(X=V_1) = p_1$	$P(X=V_2) = p_2$	….	$P(X=V_m) = p_m$

What's the smallest possible number of bits, on average, per symbol, needed to transmit a stream of symbols drawn from X's distribution? It's

$$H(X) = -p_1 \log_2 p_1 - p_2 \log_2 p_2 - \ldots - p_m \log_2 p_m$$

$$= -\sum_{j=1}^{m} p_j \log_2 p_j$$

H(X) = The entropy of X

- "High Entropy" means X is from a uniform (boring) distribution
- "Low Entropy" means X is from varied (peaks and valleys) distribution

Specific Conditional Entropy H(Y|X=v)

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Specific Conditional Entropy:

$H(Y|X=v)$ = **The entropy of** Y **among only those records in which** X **has value** v

Example:

- $H(Y|X=Math) = 1$
- $H(Y|X=History) = 0$
- $H(Y|X=CS) = 0$

Conditional Entropy H(Y|X)

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Conditional Entropy:

$H(Y|X)$ = The average specific conditional entropy of Y

= if you choose a record at random what will be the conditional entropy of Y, conditioned on that row's value of X

= Expected number of bits to transmit Y if both sides will know the value of X

$$= \Sigma_j \, Prob(X=v_j) \, H(Y \mid X = v_j)$$

Conditional Entropy

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Conditional Entropy:

$H(Y|X)$ = The average conditional entropy of Y

$= \Sigma_j Prob(X=v_j)\ H(Y \mid X = v_j)$

Example:

v_j	$Prob(X=v_j)$	$H(Y \mid X = v_j)$
Math	0.5	1
History	0.25	0
CS	0.25	0

$H(Y|X) = 0.5 * 1 + 0.25 * 0 + 0.25 * 0 = 0.5$

Information Gain

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Information Gain:

$IG(Y|X)$ = **I must transmit** Y. **How many bits on average would it save me if both ends of the line knew** X**?**

$$IG(Y|X) = H(Y) - H(Y|X)$$

Example:

- **H(Y) = 1**
- **H(Y|X) = 0.5**
- **Thus IG(Y|X) = 1 – 0.5 = 0.5**

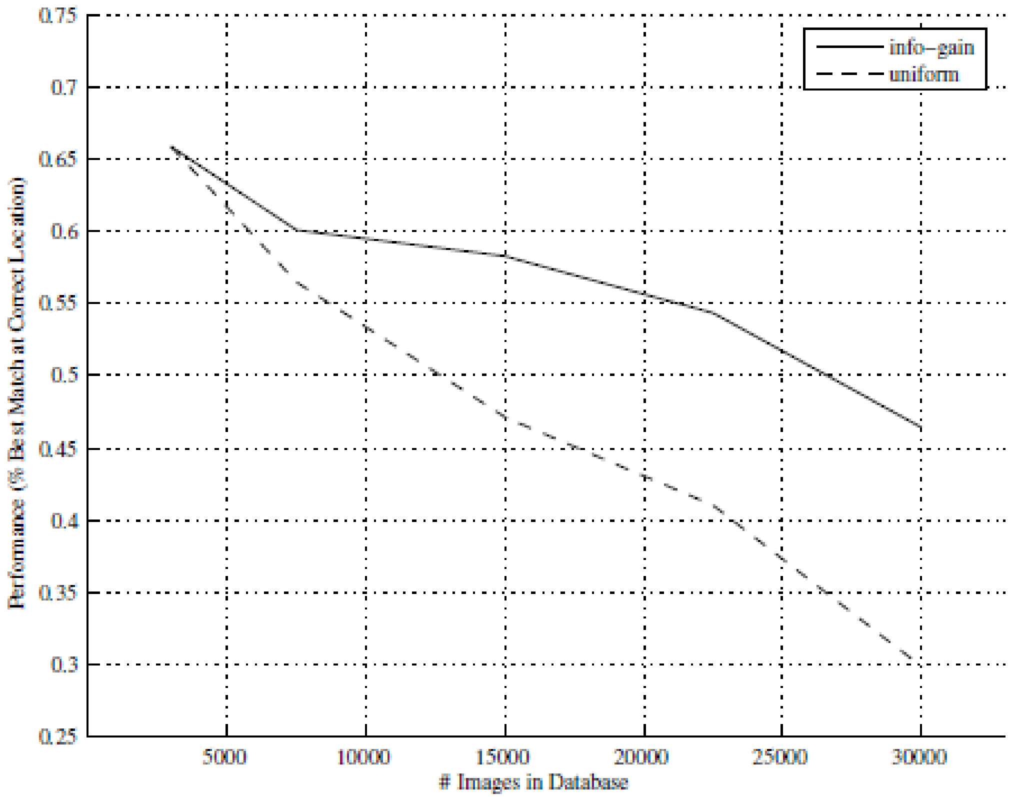
Informative Features

- $N_{W_j L_i}$ = *a*, $N_{W_j \overline{L_i}}$ = *b*
- N_{DB} **: # of images in the database**
- N_L **: # of images in each location**
- $H(L_i|W_j) =$

$$-\frac{a+b}{N_{DB}}\left[\frac{a}{a+b}\log\left(\frac{a}{a+b}\right)+\frac{b}{a+b}\log\left(\frac{b}{a+b}\right)\right]$$
$$-\frac{N_{DB}-a-b}{N_{DB}}\left[\frac{N_L-a}{N_{DB}-a-b}\log\left(\frac{N_L-a}{N_{DB}-a-b}\right)\right.$$
$$\left.+\frac{N_{DB}-N_L-b}{N_{DB}-a-b}\log\left(\frac{N_{DB}-N_L-b}{N_{DB}-a-b}\right)\right]$$

Results

- **1M VT, k=10, L=6, 7.5 million feature points**



Results

- 278 query images, 32^4 VT, 30K subset image database associated with GPS coordinate, 0.2s query time

Packing Bag-of-Features

Jegou et al.

CVPR 2009

Citation: over 27 at 2011

Binary BOF

- **Binary BOF is good for large vocabulary size**

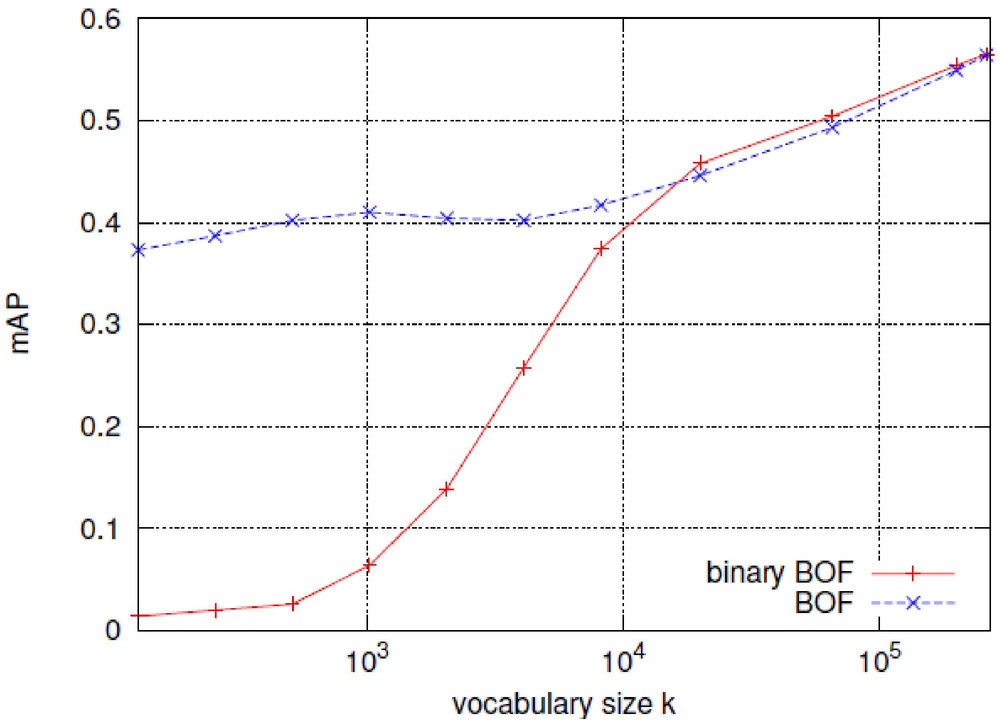


Figure 1. Search quality: BOF *vs* binary BOF

Memory Usage

- **10kb per image for raw binary BOF, 1-2kb for compressed inverted file**

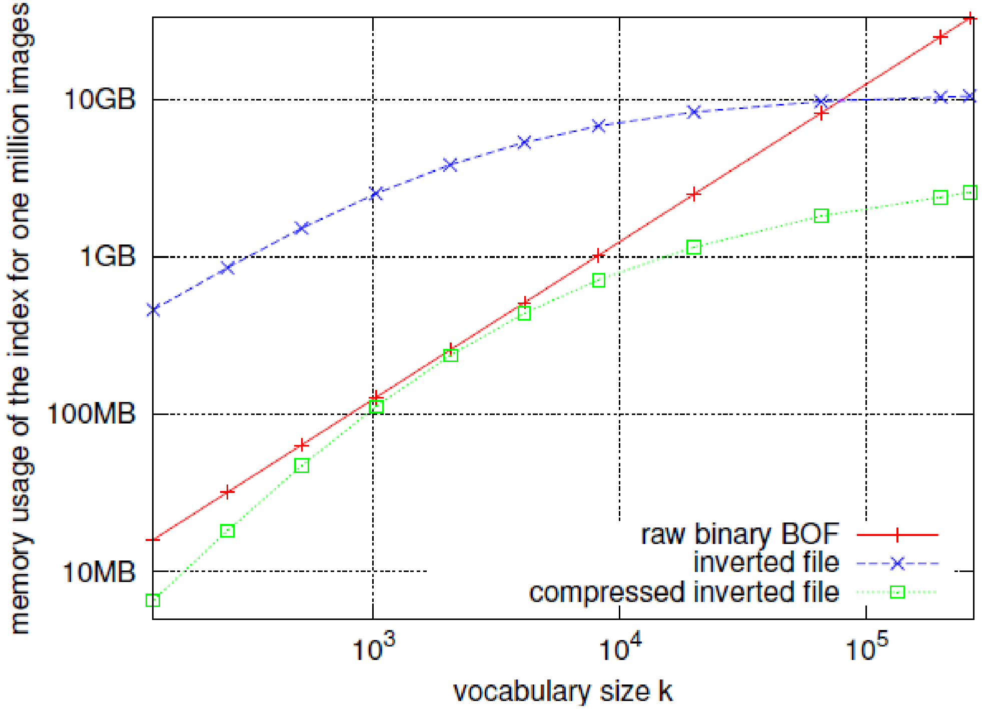
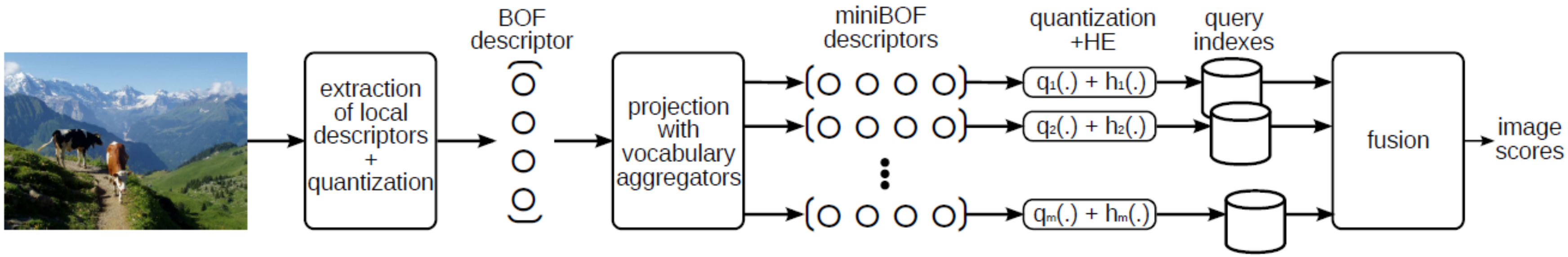


Figure 2. Binary BOF vectors: memory usage of different indexing structures for one million images.

MiniBOFs



- **Split BOF vector, project it (aggregation: dimension reduction from k to d)**

$$A_1 = \underbrace{\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}}_{k} \left.\vphantom{\begin{matrix}1\\1\\1\end{matrix}}\right\} d$$

- **Quantize each with k-means: use 4 bytes**
- **For better results, use Hamming Embedding[ECCV 2008] for each descriptors: a few more bits to encode the location with each cluster**

Results

method	k	mAP	memory usage	image hits
BOF	1k	0.414	3,087	1,484
BOF	20k	0.446	10,364	1,471
BOF	200k	0.549	12,886	1,412
binary BOF	20k	0.458	8,291	1,471
binary BOF	200k	0.554	10,309	1,412
compressed binary BOF*	20k	0.458	1,174	1,471
compressed binary BOF*	200k	0.554	1,830	1,412
miniBOF, m=1	1k	0.255	20	19
miniBOF, m=4	1k	0.368	80	48
miniBOF, m=8	1k	0.403	160	68
miniBOF, m=16	1k	0.426	320	93
miniBOF, m=32	1k	0.452	640	120

Table 1. Comparison of the different BOF approaches on the Holidays dataset: search quality (mAP), memory usage (bytes per database image), and average number of image hits per query image. The hits values should be compared to the total number of images (1491). m is the number of miniBOFs; *estimation based on the binary BOF vector entropy.

Achieves about 2 times lower memory given the similar mAP

Improve its quality by using multiple BoF, while keeping memory low

Next Time…

- **Novel applications**