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

Josef Sivic and Andrew Zisserman

Goal

- Google search for videos
- Query is an portion of a frame of a video selected by the user

Google Text Search

- Web pages are parsed into words
- Words are replaced by their root word
- Stop list to filter common words
- Remaining words represent that web page as a vector weighted based on word frequency

Text Retrieval

- Efficient retrieval for with an index
- Text is retrieved by computing its vector of word frequencies, return documents with the closest vectors
- Consider order and location of words

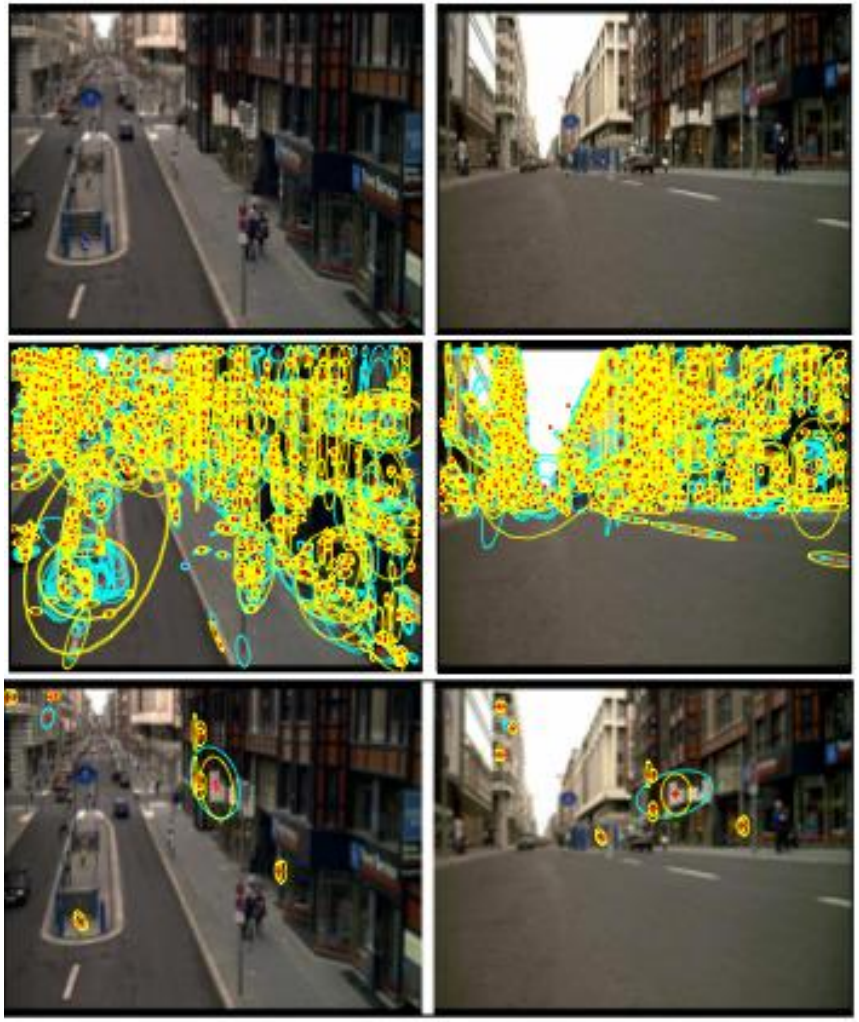
Approach

- Apply text search properties to image search

Video Google: Descriptors

- Compute two types of covariant regions: Shape Adapted and Maximally Stable
- Regions computed in grayscale

Descriptors



Descriptors

- Each elliptical region is then represented by a SIFT descriptor
- Descriptor is averaged over the frames the region exists in
- Reduce noise: filter regions which do not exist in more than 3 frames
- Reject 10% of the regions with the largest diagonal covariance matrix

Build “Visual Words”

- Quantize the descriptors into visual words for text retrieval
- 1000 regions per frame and 128-vector descriptor
- Select 48 scenes containing 10,000 frames
- 200K descriptors

Clustering descriptors

- K-means clustering
- Run several times with random initial assignments
- $D(x1, x2) = \mathrm{sqrt}((x1 - x2)^T \sum^{-1}(x1 - x2))$
- MS and SA regions are clustered separately

Indexing using text retrieval methods

- Term frequency - inverse document frequency used for weighting the words of a document

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

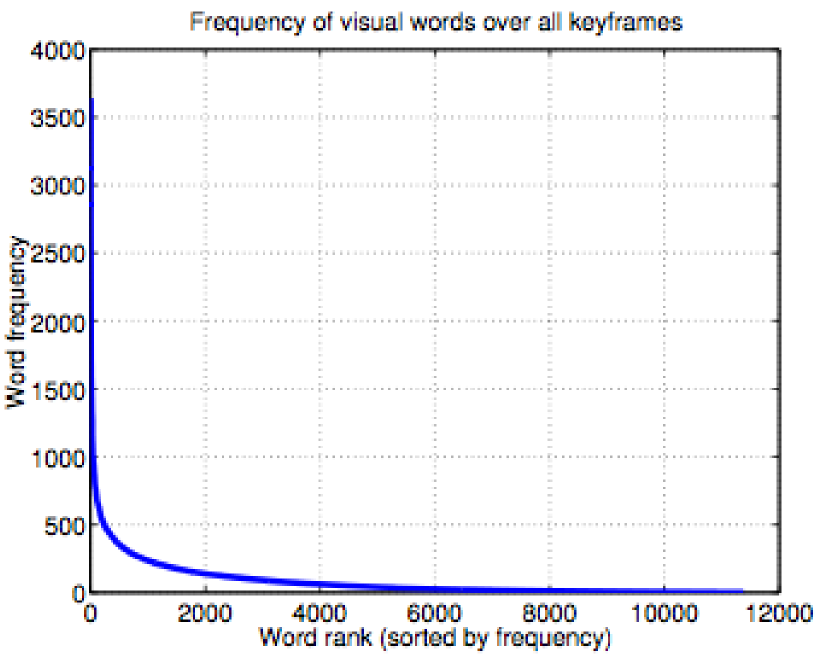
- Retrieval: documents are ranked by their normalized scalar product between the query vector and all the document vectors

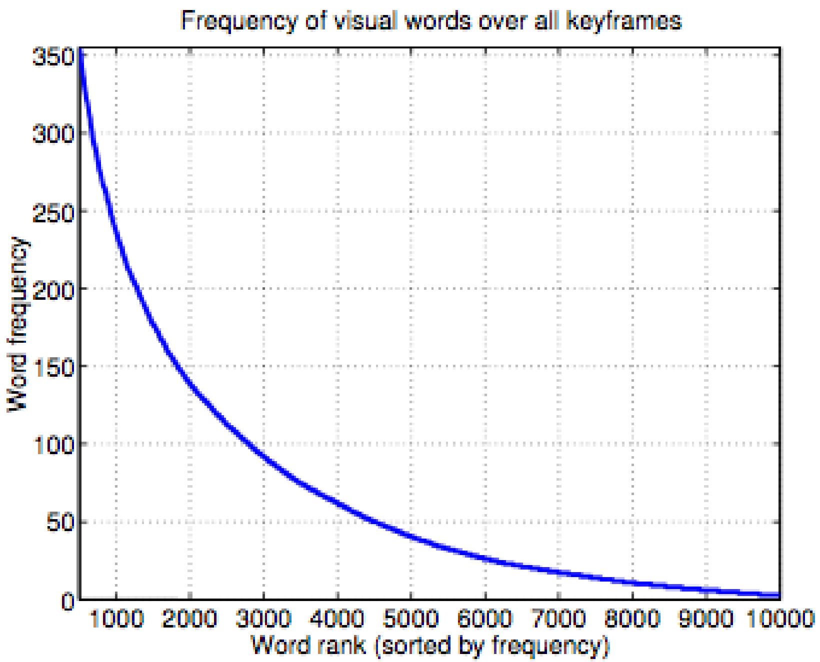
Image Retrieval

- Video google: The visual words of the query are the visual words in the user-specified portion of a frame
- Search the index with the visual words to find all the frames which contain the same word
- Rank all the results, return the most relevant results

Stop List

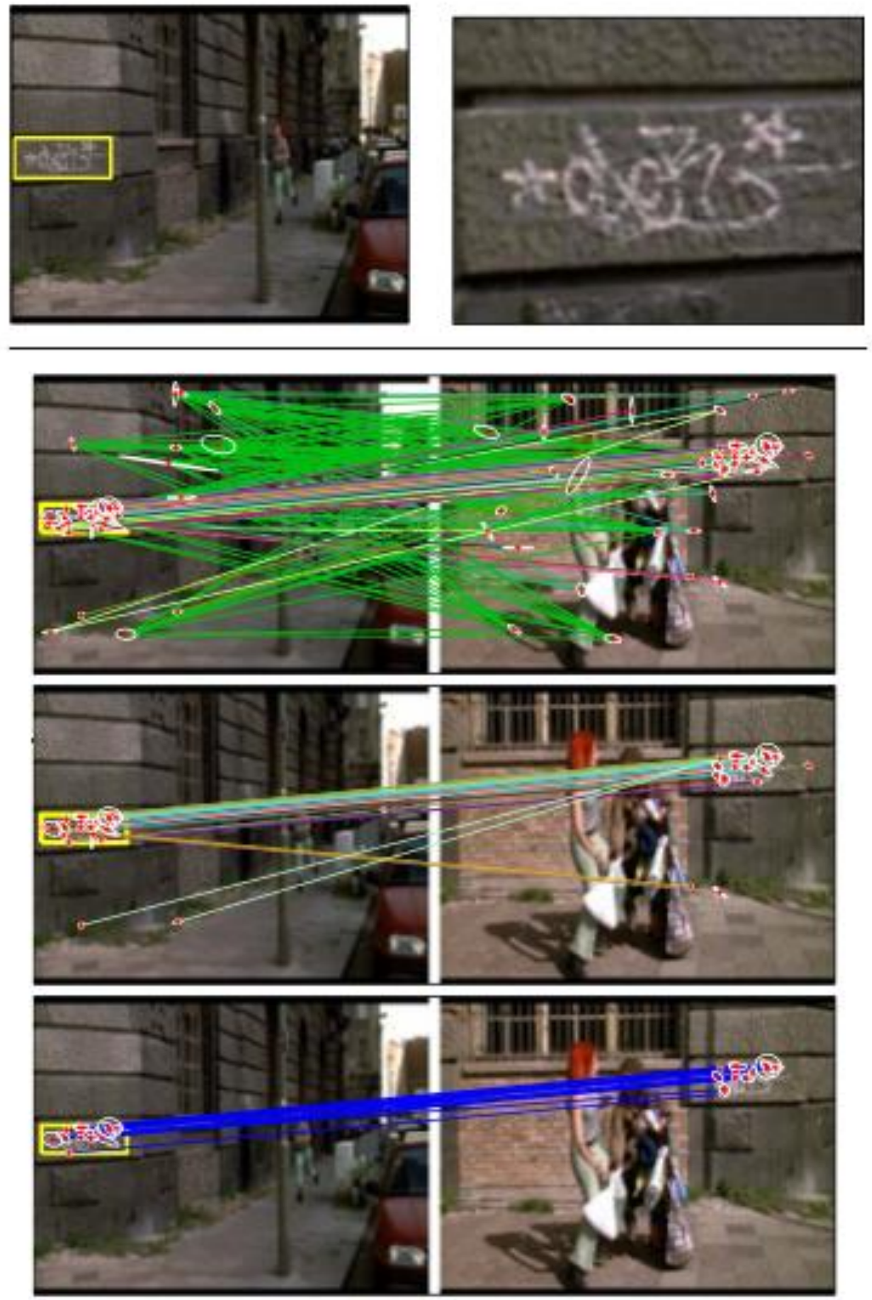
- Visual words in the top 5% and bottom 10% are stopped





Spatial Consistency

- Google increases the ranking of documents where the query words appear close together in the searched text
- In video: 15 nearest neighbors defines search area
- Regions in this area by the query region vote on each match
- Re-ranked on the number of votes

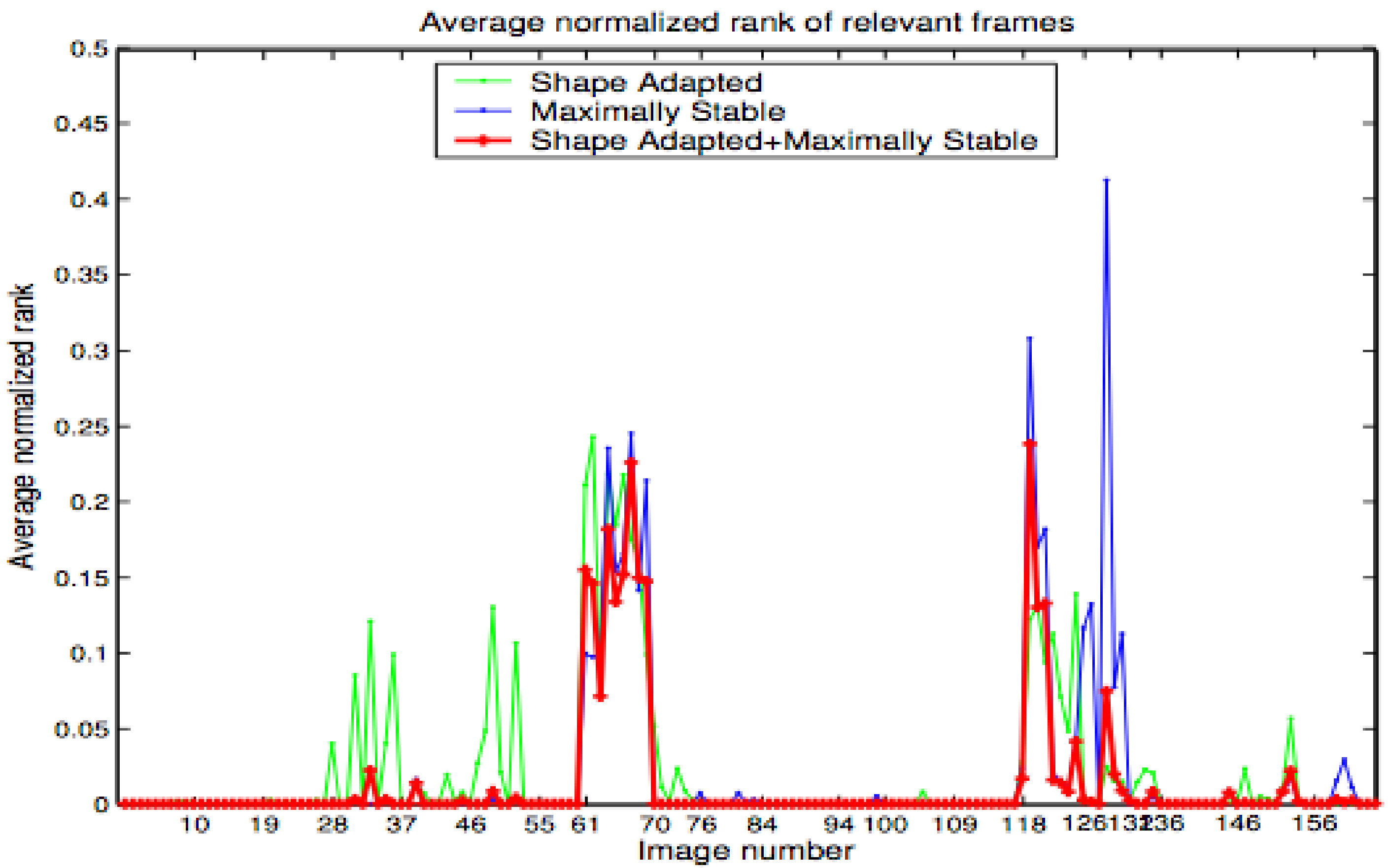


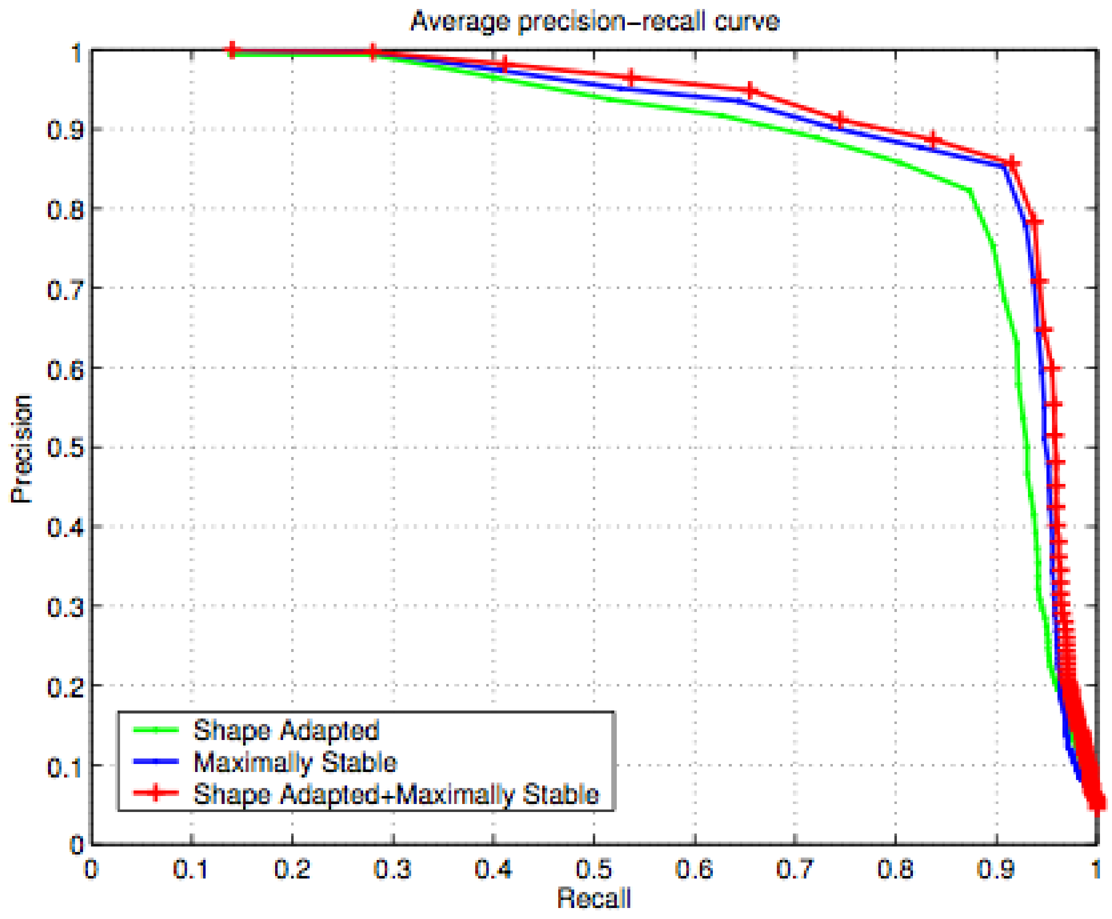
Evaluation

- Tested on feature length movies with 100K - 150K frames
- Use one frame per second
- Ground truth determined by hand
- Retrieval performance measured by averaged rank of relevant images

$$\widetilde{Rank} = \frac{1}{NN_{rel}}\left(\sum_{i=1}^{N_{rel}} R_i - \frac{N_{rel}(N_{rel}+1)}{2}\right)$$

	binary	*tf*	*tf-idf*
SA	0.0265	0.0275	0.0209
MS	0.0237	0.0208	0.0196
SA+MS	0.0165	0.0153	0.0132


Average normalized rank of relevant frames
Shape Adapted
Maximally Stable
Shape Adapted+Maximally Stable
Average normalized rank
Image number
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
10
19
28
37
46
55
61
70
76
84
94
100
109
118
126
146
156



Average precision−recall curve
Precision
Recall
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Shape Adapted
Maximally Stable
Shape Adapted+Maximally Stable

Example

Questions?

Scalable Recognition with a Vocabulary Tree

David Nister and

Henrik Stewenius

Vocabulary Tree

- Continuation of Video google
- 10,000 visual words in the database
- Offline crawling stage to index video takes 10 seconds per frame

Vocabulary Tree

- Too slow for a large database
- Larger databases result in better retrieval quality
- More words utilizes the power of the index: less database images must be considered
- On the fly insertion of new objects into the database

Training

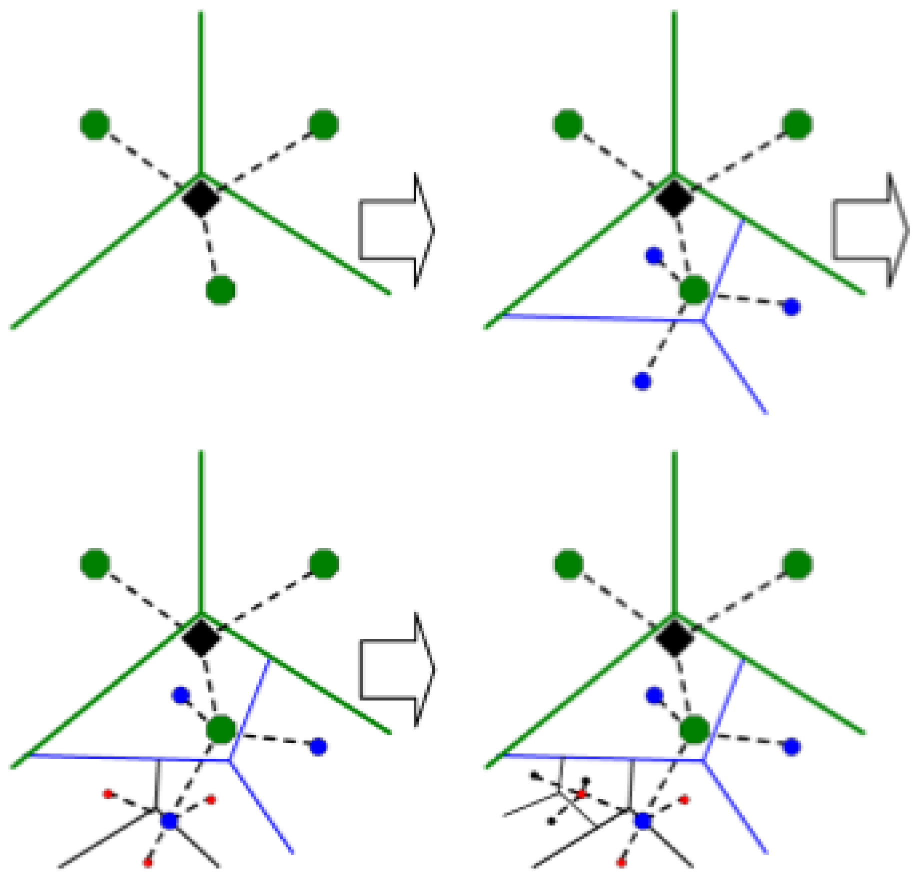
- Training with hierarchical k-means
- More efficient than k-means
- 35,000 training frames instead of 400 with video google

Feature Extraction

- Maximally Stable regions used only
- Build SIFT descriptor from the region

Building Vocab Tree

- Hierarchical k-means, with k being the number of children nodes
- First run k-means to find k clusters
- Recursively apply to each cluster L times
- Visual words become the nodes



Performance

- Increasing the size of the vocabulary is logarithmic
- K = 10, L = 6: one million leaf nodes

Retrieval

- Determine the visual words from the query
- Propagate the region descriptor down the tree selecting the closest cluster at each level

Scoring

- Determine the relevance of a query image to a database image based on the similarity of their paths down the tree
- Use TD-IDF to assign weights to the query and database image vector

Scoring

- Use TD-IDF for weights of descriptor vectors
- Normalized relevance score:

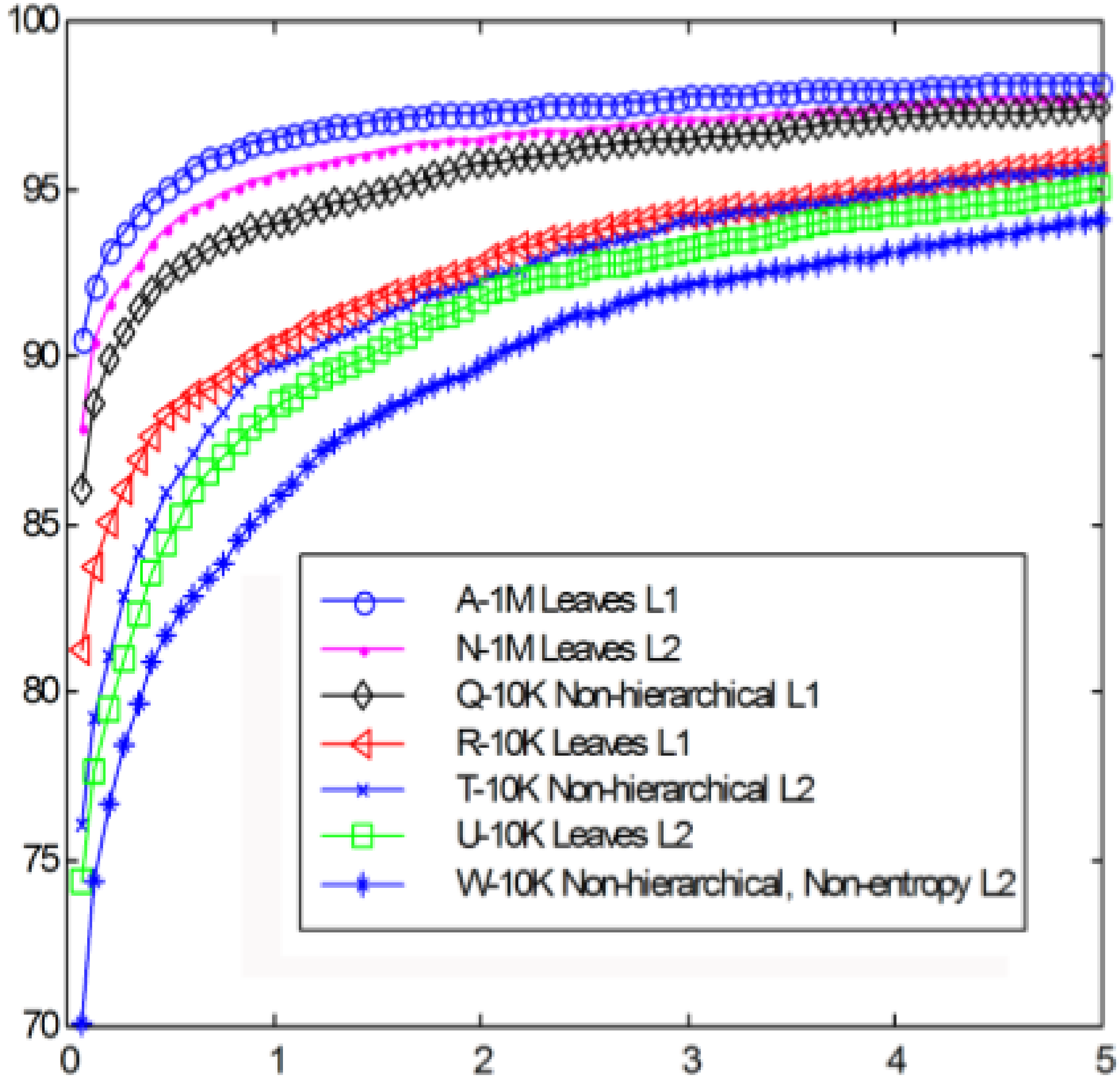
$$s(q,d) = \left\| \frac{q}{\| q \|} - \frac{d}{\| d \|} \right\|$$

- L_1-normalization is the most effective

Results

- Tested on a ground truth database of 6,376 images
- Groups of four images of the same object

Results



100
95
90
85
80
75
70
0
1
2
3
4
5
A-1M Leaves L1
N-1M Leaves L2
Q-10K Non-hierarchical L1
R-10K Leaves L1
T-10K Non-hierarchical L2
U-10K Leaves L2
W-10K Non-hierarchical, Non-entropy L2

Results

- Tested on a database of 1 million images of CD covers
- Sub-second retrieval times for a database of a million images
- Performance increases with the number of leaf nodes

Me	En	No	S%	Voc-Tree	Le	Eb	Perf
A	**y/y**	**L1**	**0**	**6x10=1M**	**1**	**ir**	**90.6**
B	y/y	L1	0	6x10=1M	1	vr	90.6
C	y/y	L1	0	6x10=1M	2	ir	90.4
D	n/y	L1	0	6x10=1M	2	ir	90.4
E	y/n	L1	0	6x10=1M	2	ir	90.4
F	n/n	L1	0	6x10=1M	2	ir	90.4
G	n/n	L1	0	6x10=1M	1	ir	90.2
H	y/y	L1	m2	6x10=1M	1	ir	90.0
I	y/y	L1	0	6x10=1M	3	ir	89.9
J	y/y	L1	0	6x10=1M	4	ir	89.9
K	y/y	L1	0	6x10=1M	2	vr	89.8
L	y/y	L1	0	6x10=1M	2	ip	89.0
M	y/y	L1	m5	6x10=1M	1	ir	89.1
N	**y/y**	**L2**	**0**	**6x10=1M**	**1**	**ir**	**87.9**
O	y/y	L2	0	6x10=1M	2	ir	86.6
P	y/y	L1	110	6x10=1M	2	ir	86.5
Q	**y/y**	**L1**	**0**	**1x10K=10K**	**1**	**-**	**86.0**
R	**y/y**	**L1**	**0**	**4x10=10K**	**2**	**ir**	**81.3**
S	y/y	L1	0	4x10=10K	1	ir	80.9
T	**y/y**	**L2**	**0**	**1x10K=10K**	**1**	**-**	**76.0**
U	**y/y**	**L2**	**0**	**4x10=10K**	**1**	**ir**	**74.4**
V	y/y	L2	0	4x10=10K	2	ir	72.5
W	**n/n**	**L2**	**0**	**1x10K=10K**	**1**	**-**	**70.1**

Questions?