Bag of Visual Words for Finding Similar Images

By Fiseha B. Tesem, PhD

Recap

- Introduction to Object Recognition

Outline

- Bag of Visual Words

What is Bag of Visual Word for?

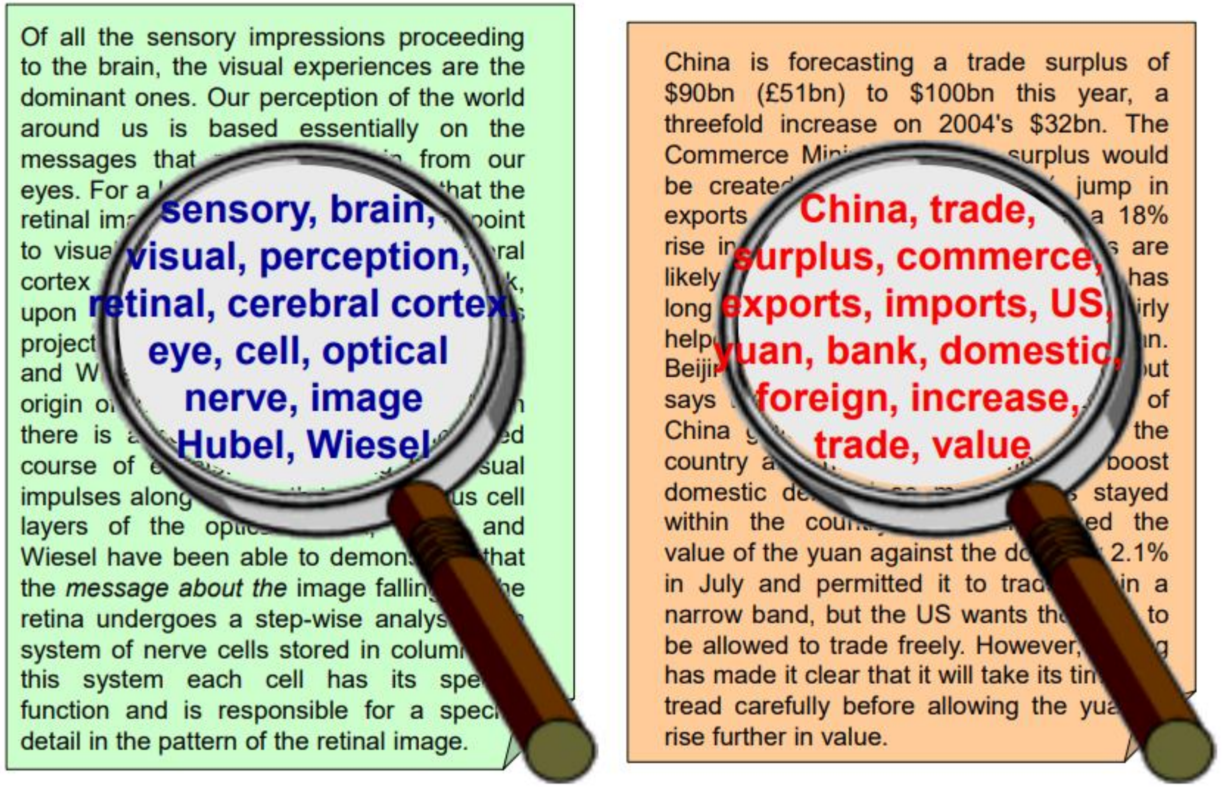
- Finding images in a database, which are similar to a given query image.
  - E.g. Google image search
- Computing image similarities
- Compact representation of images

Why Bag of words?

Origin 1: Analogy to Text Documents

- Orderless document representation: frequencies of words from dictionary

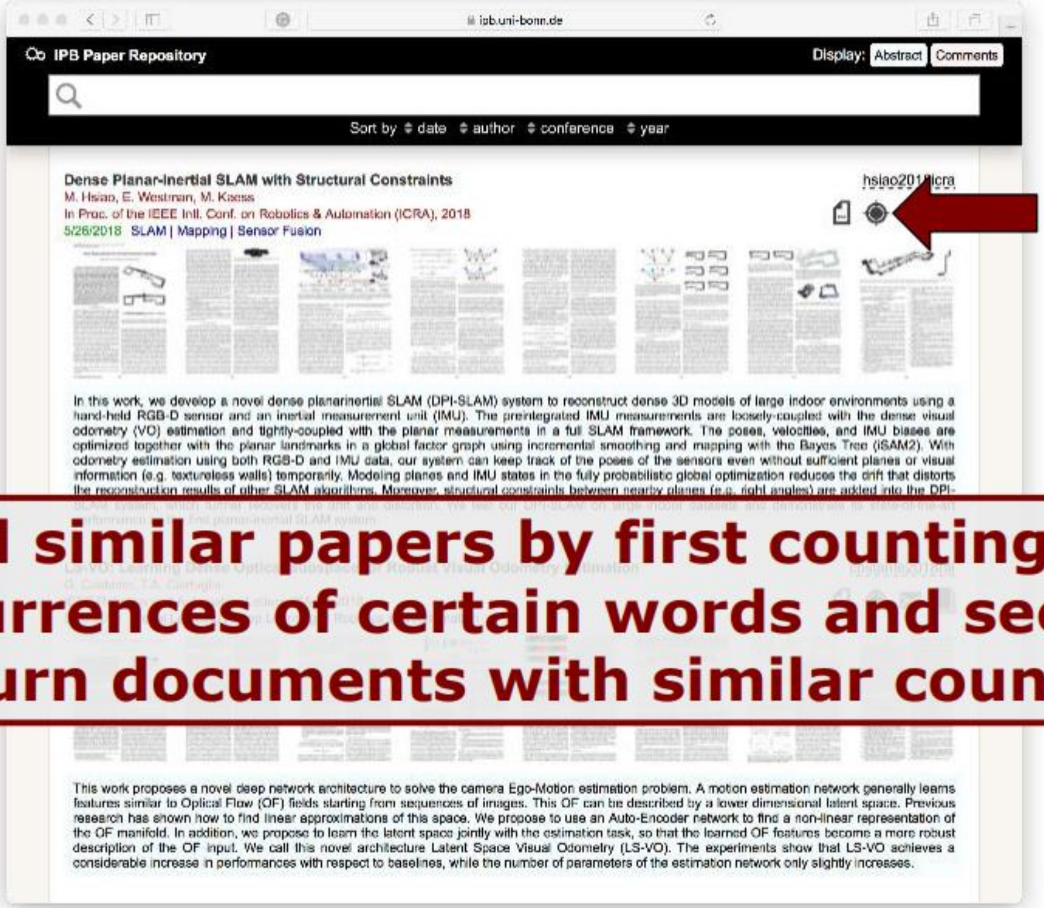
Salton & McGill (1983)



- Reduce the documents into word frequencies. Based on the frequencies:
  - Application
    - Group document based on the similarities:
      - E.g. Legal, reports, trade, sports …
    - Find documents similar to a give query

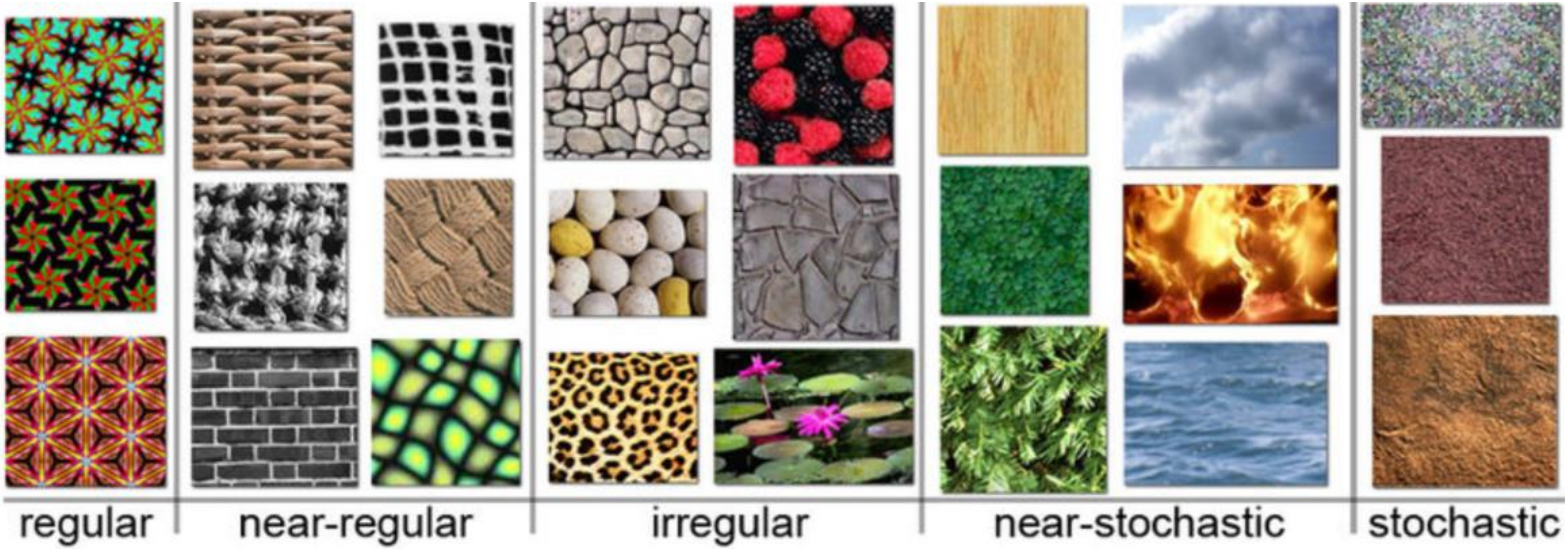
[Image courtesy: Fei-Fei Li]

Looking for similar Papers



"find similar papers by first counting the occurrences of certain words and second return documents with similar counts."

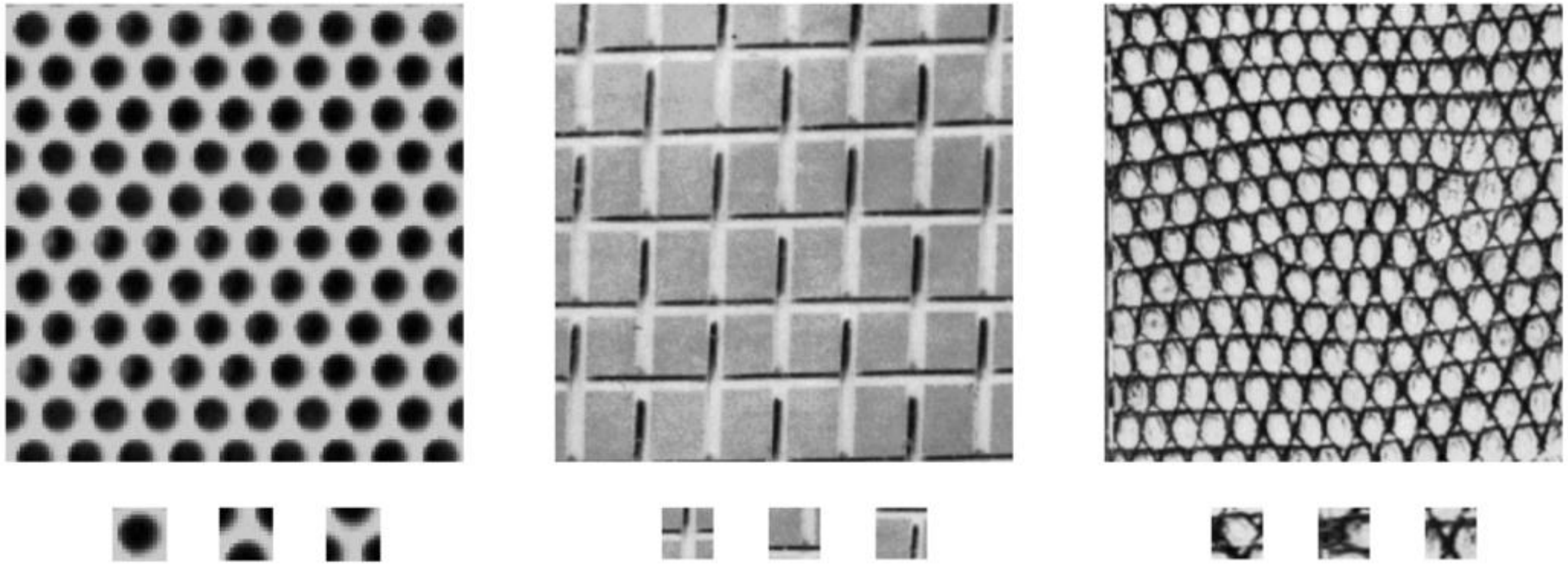
Origin 2:Texture Recognition



Example textures (from Wikipedia)

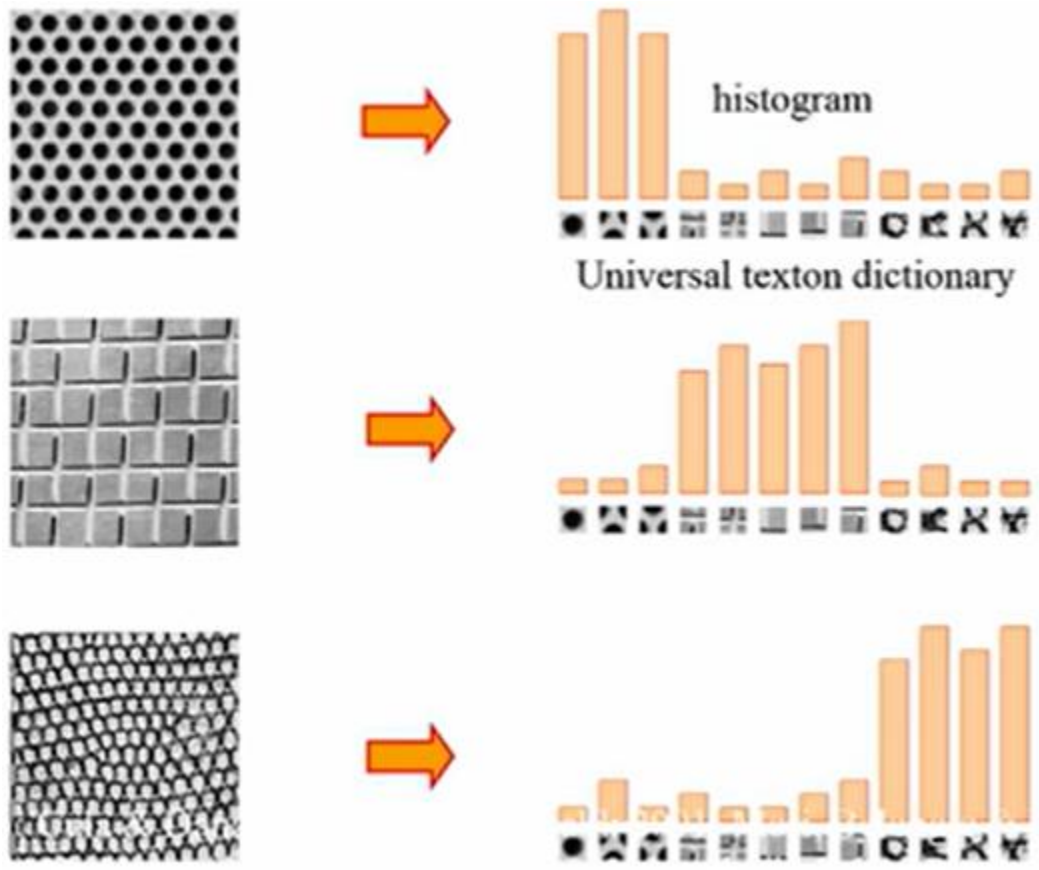
Origin 2:Texture Recognition

- Texture is characterized by the repetition of basic elements or **textons**.



Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001; Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

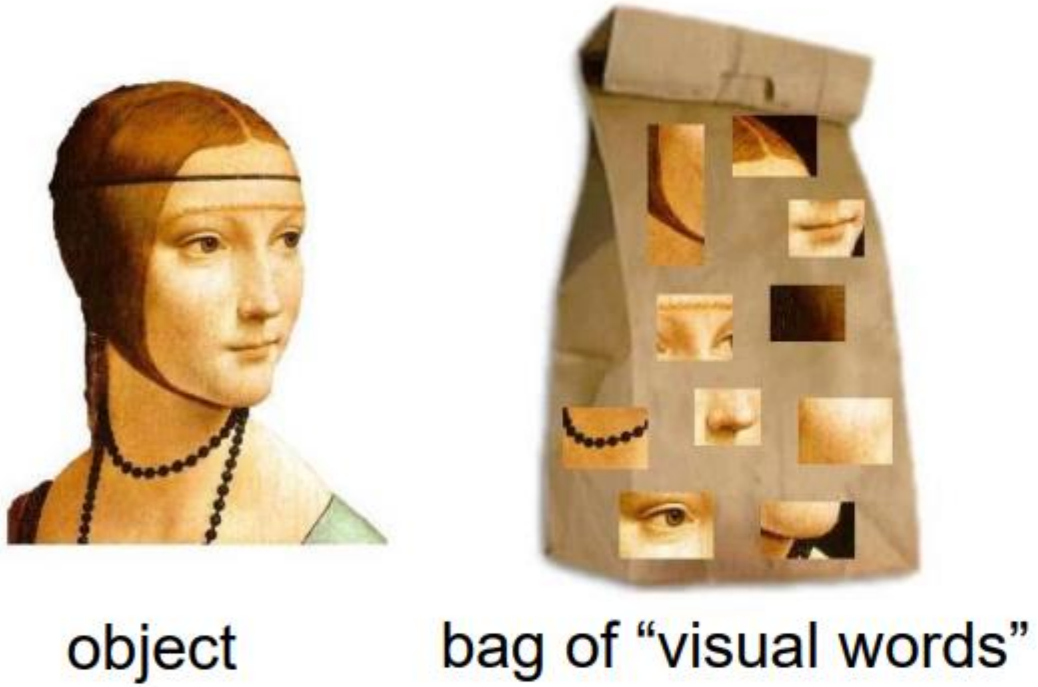
Origin 2:Texture Recognition



Brief Summary how BoW operates

Brief Summary: how BoW operates?

- Analogy to documents: The content of a can be inferred from the frequency of relevant words that occur in a document



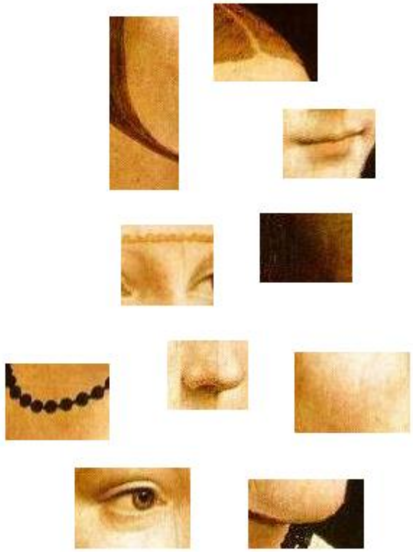
[Image courtesy: Fei-Fei Li]

Bag of (Visual) Words

1

- Convert pixel information to visual words using **feature descriptor techniques**. e.g. SIFT
- Visual words = independent features
- Breakdown the image into independent features

face



features

[Image courtesy: Fei-Fei Li]

Bag of (Visual) Words

2

- Which of these words allowed to describe best the image?
  - Construct a dictionary of representative words
  - Use only words from the dictionary

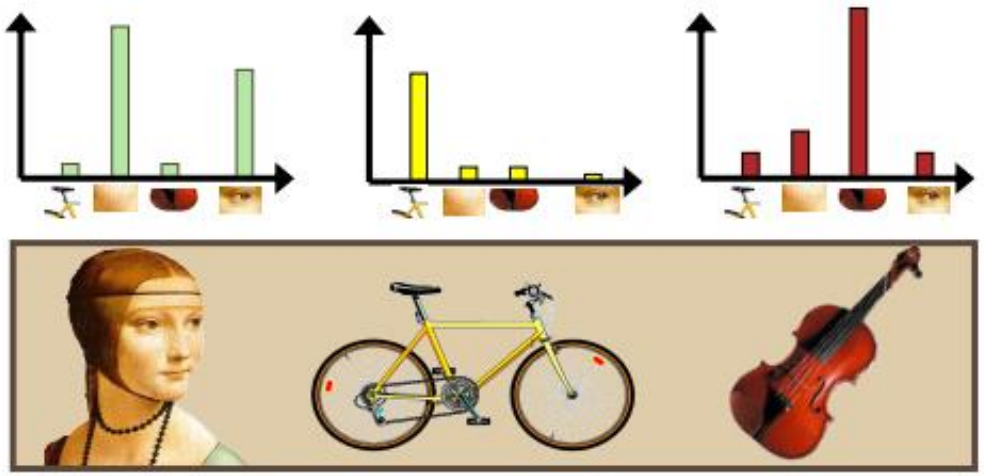
dictionary (“codebook“)

[Image courtesy: Fei-Fei Li]

Bag of (Visual) Words

3

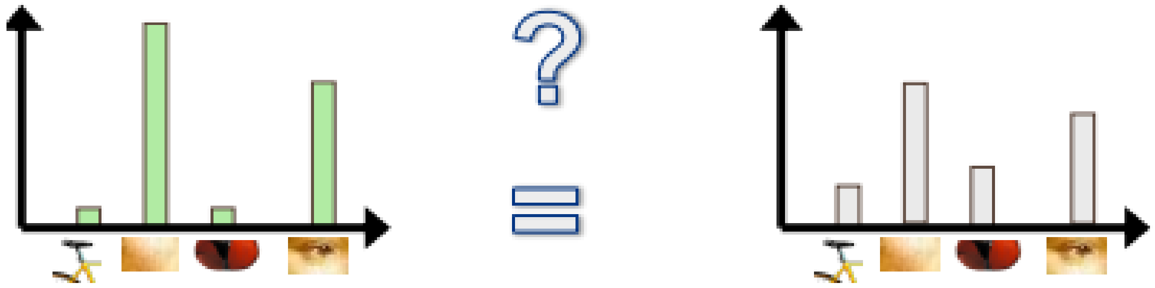
- Represent the images based on a histogram of word occurrences.



[Image courtesy: Fei-Fei Li]

Bag of (Visual) Words

- How can I make a decision if two image are similar?
  - Visual words = independent features
  - Words from the dictionary
  - Represent the images based on a histogram of word occurrences
  - Image comparisons are performed based on such word histograms



Is the distribution of visual words similar? Use distance metrics or similarity.

[Image courtesy: Fei-Fei Li]

Overview: From Images to Histograms

- E.g. Visual Place recognition

[Image courtesy: Olga Vysotska]

Overview: Input Image

[Image courtesy: Olga Vysotska]

Overview: Extract Features

- Extract Features
  - Let say we use SIFT
- Assume we have dictionary (we will discuss later how to build it)
  - Take this feature and assign them to the closest word in our dictionary

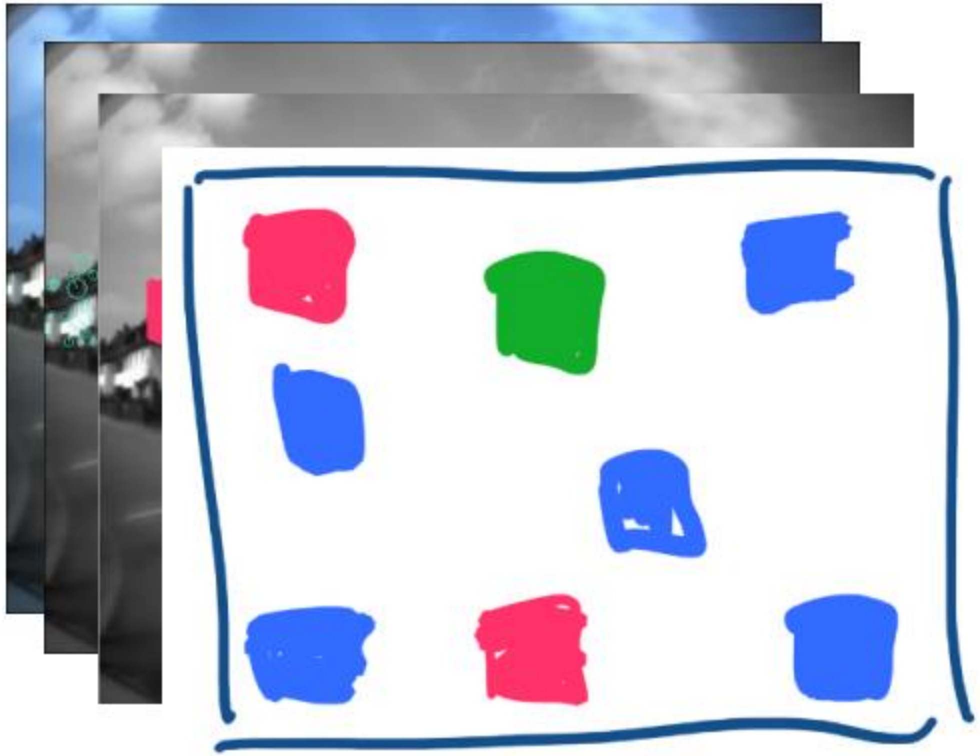
[Image courtesy: Olga Vysotska]

Overview: Visual Words

- Reduce feature to visual words.
- Mapped SIFT features onto Visual words
- Every word represent similar looking features

[Image courtesy: Olga Vysotska]

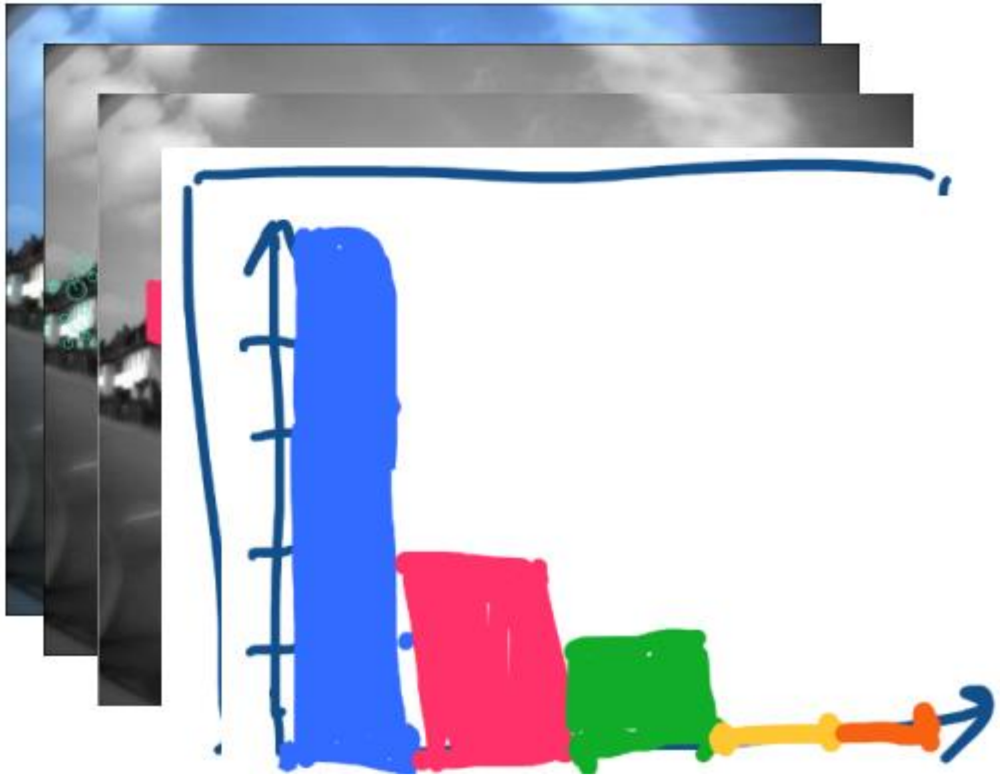
Overview: No Pixel Values



- After all, no need to consider images.
- No pixel values

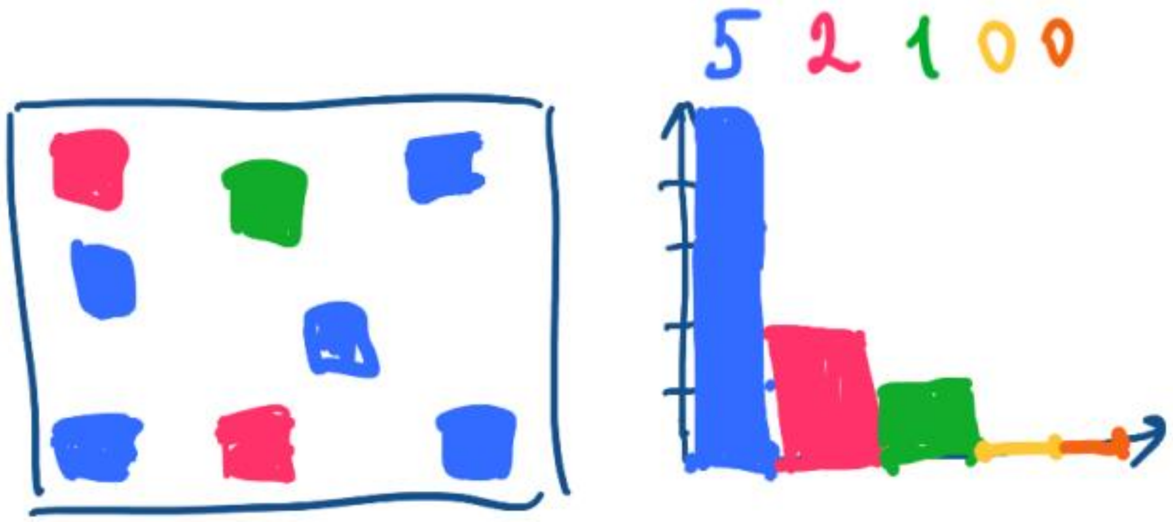
[Image courtesy: Olga Vysotska]

Overview: Word Occurrences



[Image courtesy: Olga Vysotska]

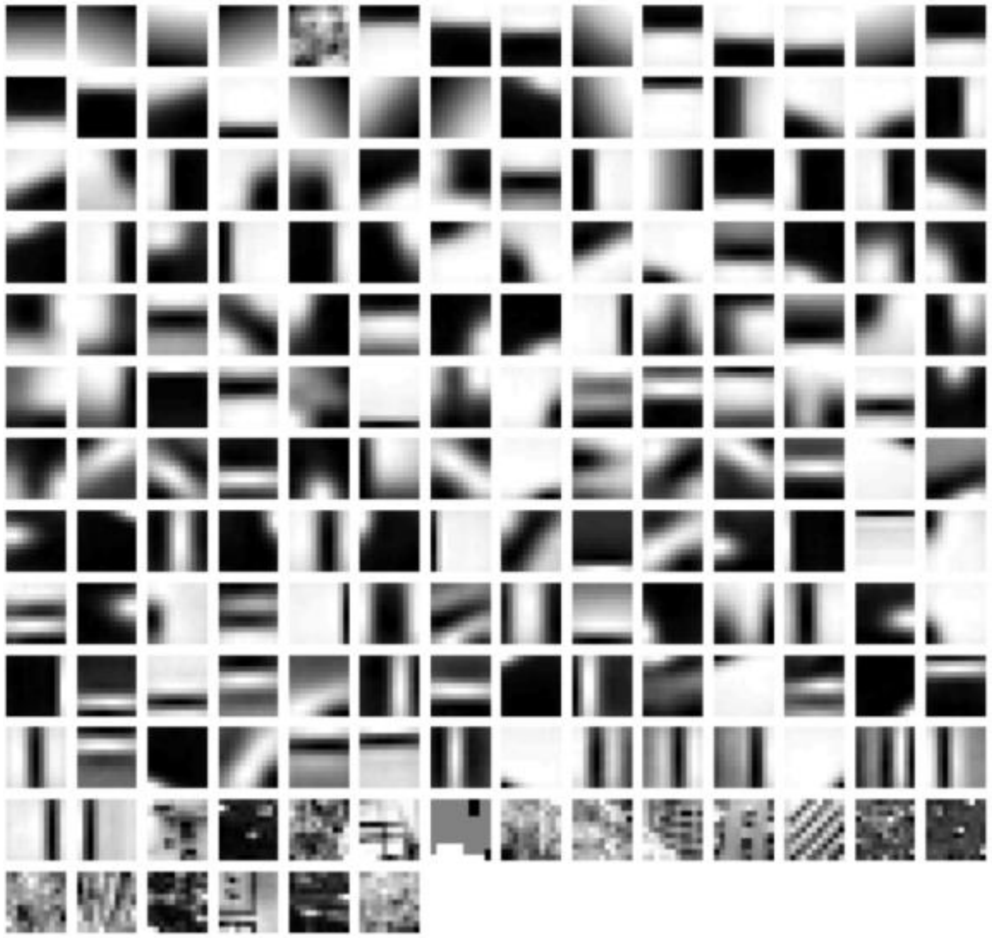
Images to Histograms



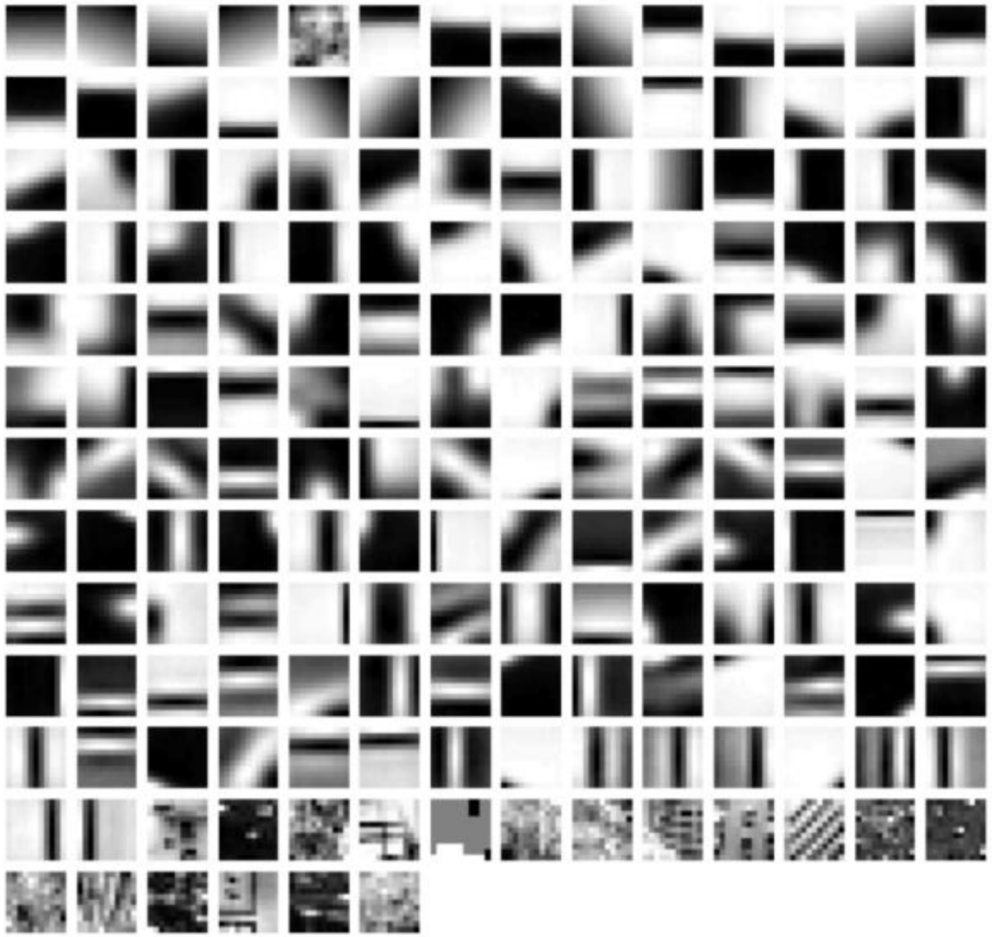
- This histogram can be expressed by vector [5,2,1,0,0].
  - But the order should be the same for all images

[Image courtesy: Olga Vysotska]

Example of Visual words/Vocabulary



Where Do the Visual Words Come From?



[Partial image courtesy: Fei-Fei Li]

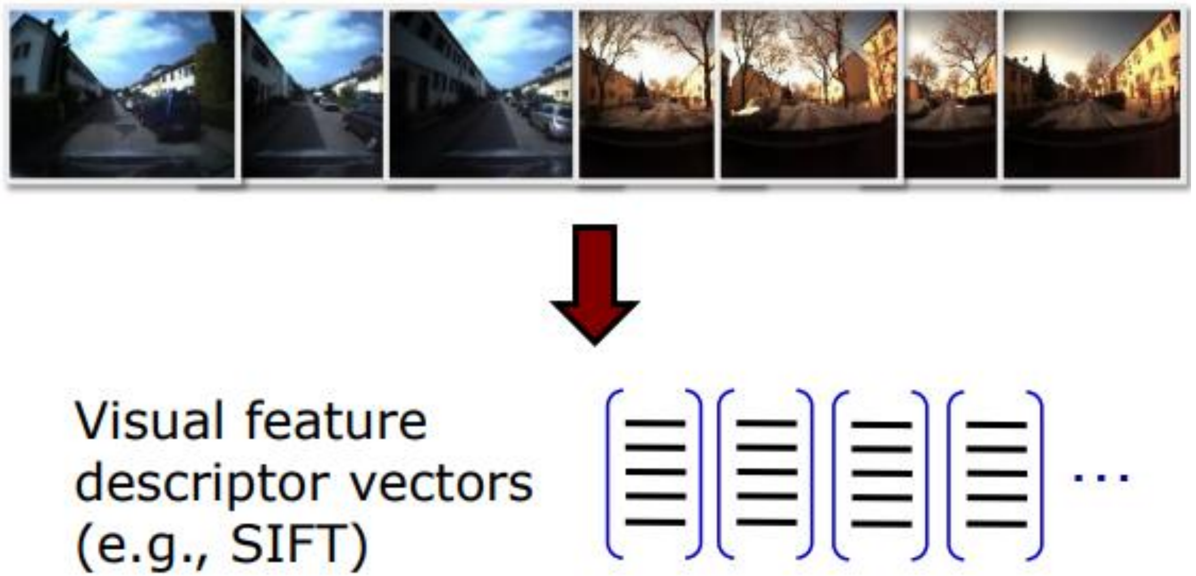
Dictionary

- A dictionary defines the list of words that are considered
- The dictionary defines the x-axes of all the word occurrence histograms
- The dictionary must remain fixed/**Is learned once.**

- **The dictionary is typically learned from a Large database.**
- **We don’t specify it by hand/manually.**

How can we do that?

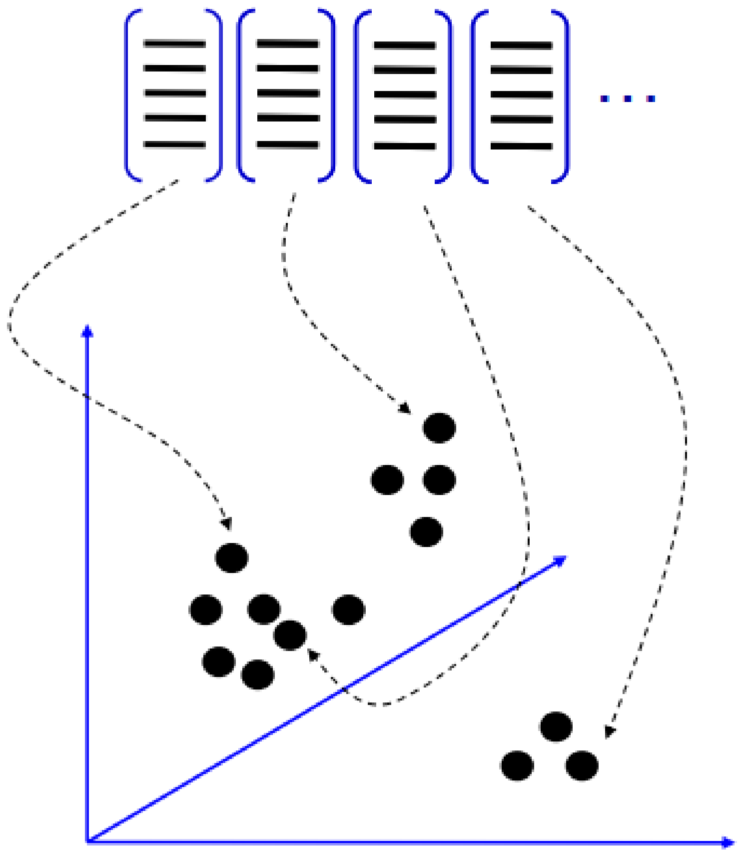
Extract Feature Descriptors from a Training Dataset



- Assume you have training database.
  - The dataset should be something related to our problem/task.
  - No need to label it
  - We will use **unsupervised learning** approach.
    - E.g SIFT to extract visual features for each images.

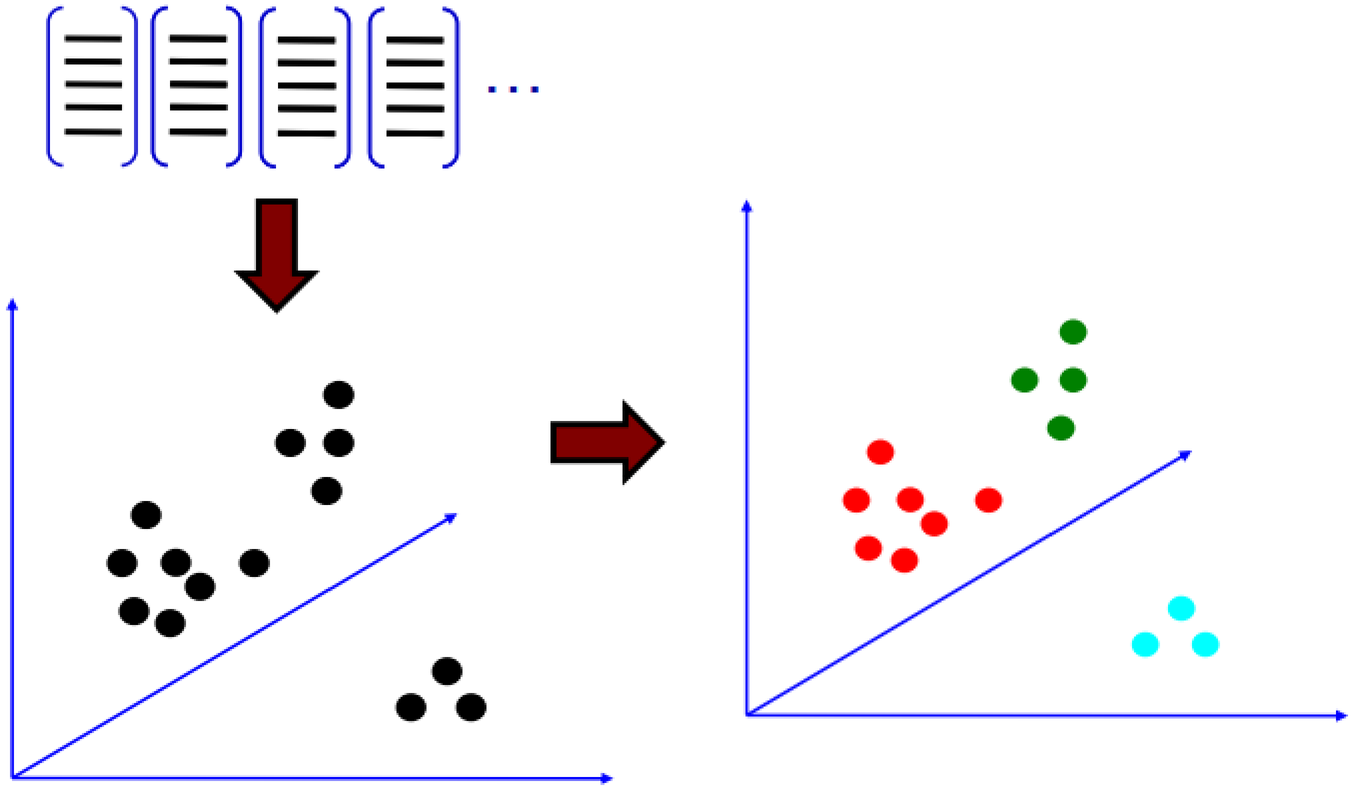
[Partial image courtesy: Fei-Fei Li]

Feature Descriptors are Points in a High-Dimensional Space



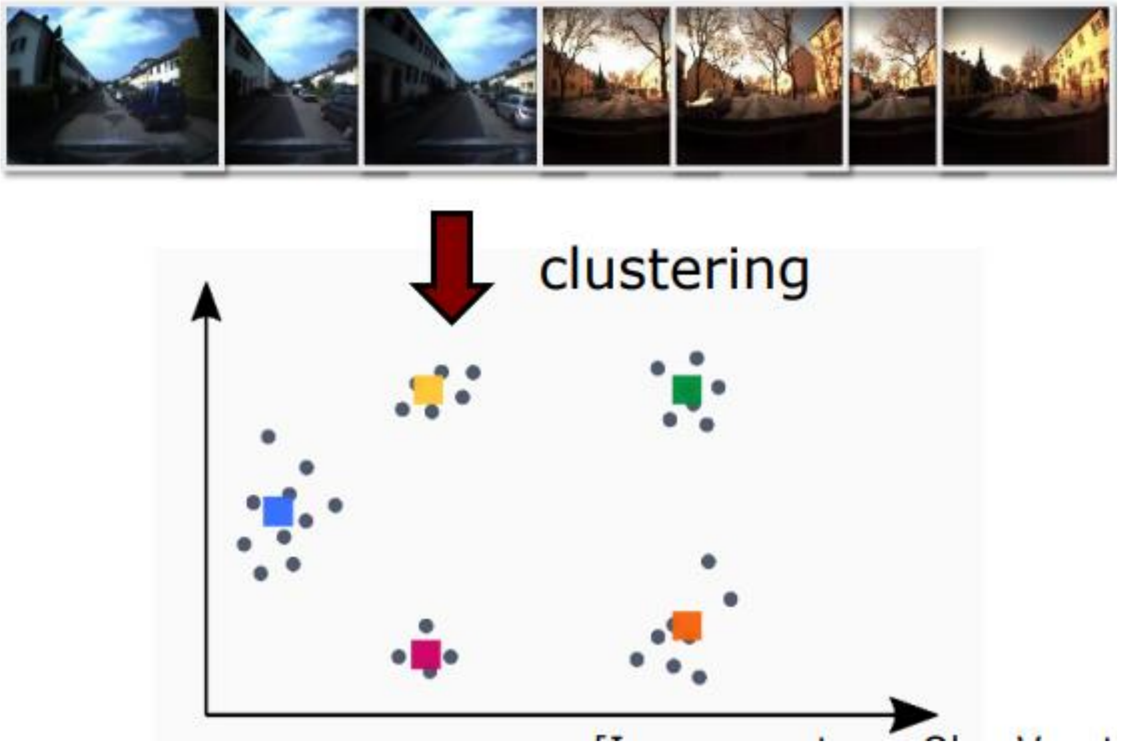
[Partial image courtesy: Fei-Fei Li]

Group Similar Descriptors



[Image courtesy: Fei-Fei Li]

Clusters of Descriptors from Data Forms the Dictionary



- Use any clustering algorithm,

[Image courtesy: Olga Vysotska]

K-Means Clustering

K-Means Clustering: overview

- Partitions the data into k clusters
- Clusters are represented by centroids
- A centroid is the mean of data points

- Objective:
  - Find the k cluster centers and assign the data points to the nearest one, such that the squared distances to the cluster centroids are minimized

K-Means Clustering for Learning the BoVW Dictionary

- Partitions the features into k groups
- The centroids form the dictionary/Visual words.
- Features will be assigned to the closest centroid (visual word)

- Approach:
  - Find k word and assign the features to the nearest word, such that the squared distances are minimized

K-Means Clustering (Informally)

- Initialization: Choose k arbitrary centroids as cluster representatives
- Repeat until convergence
  - Assign each data point to the closest centroid
  - Re-compute the centroids of the clusters based on the assigned data points

K-Means Algorithm

Initialize $\boldsymbol{m}_i, i = 1, \ldots, k$, for example, to k random $\boldsymbol{x}^t$

Repeat

For all $\boldsymbol{x}^t \in \mathcal{X}$

$$b_i^t \leftarrow \begin{cases} 1 & \text{if } \|\boldsymbol{x}^t - \boldsymbol{m}_i\| = \min_j \|\boldsymbol{x}^t - \boldsymbol{m}_j\| \\ 0 & \text{otherwise} \end{cases}$$

For all $\boldsymbol{m}_i, i = 1, \ldots, k$

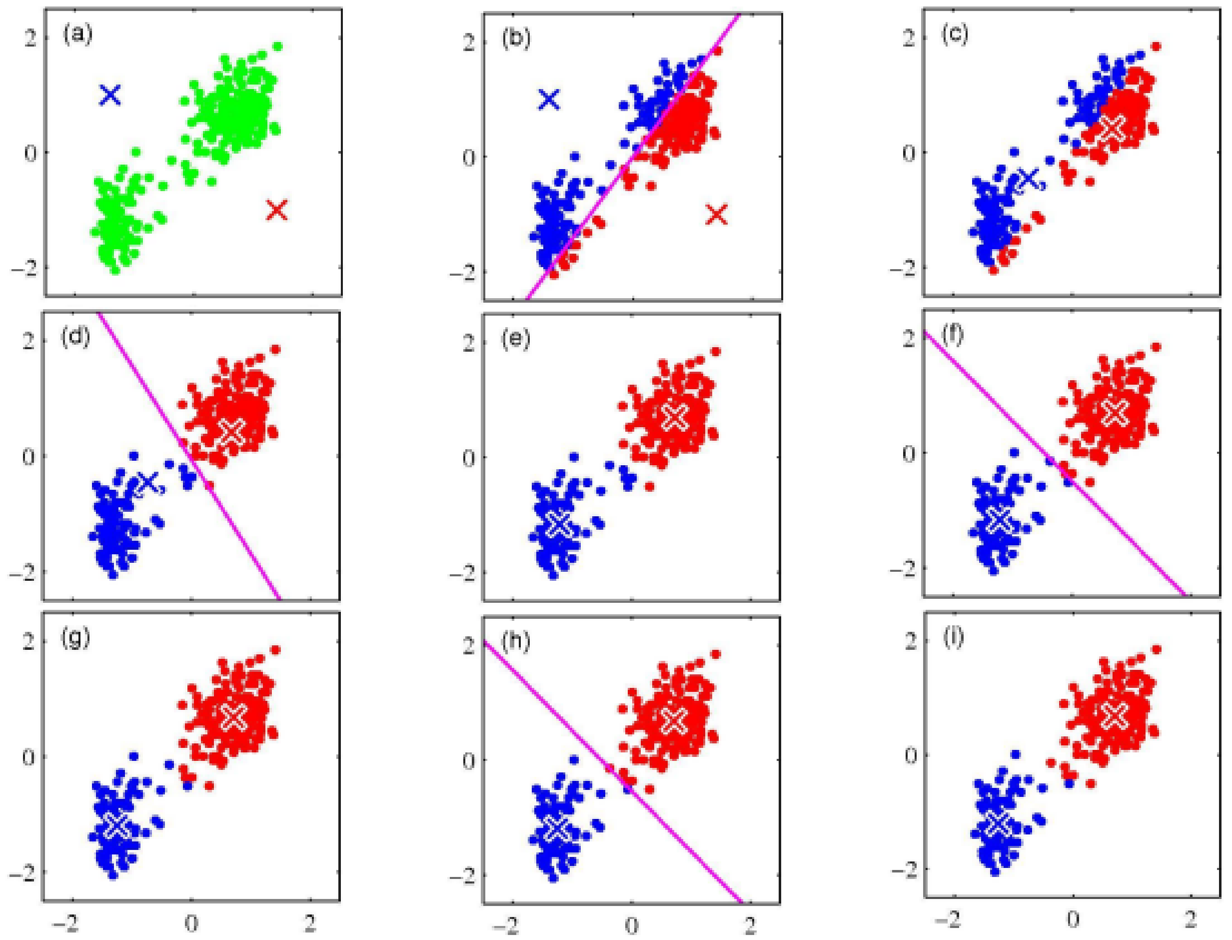
$$\boldsymbol{m}_i \leftarrow \sum_t b_i^t \boldsymbol{x}^t / \sum_t b_i^t$$

Until $\boldsymbol{m}_i$ converge

Re-compute the cluster means using the current cluster memberships

Assign each data point to the closest cluster

K-Means Example



[Image courtesy: Bishop]

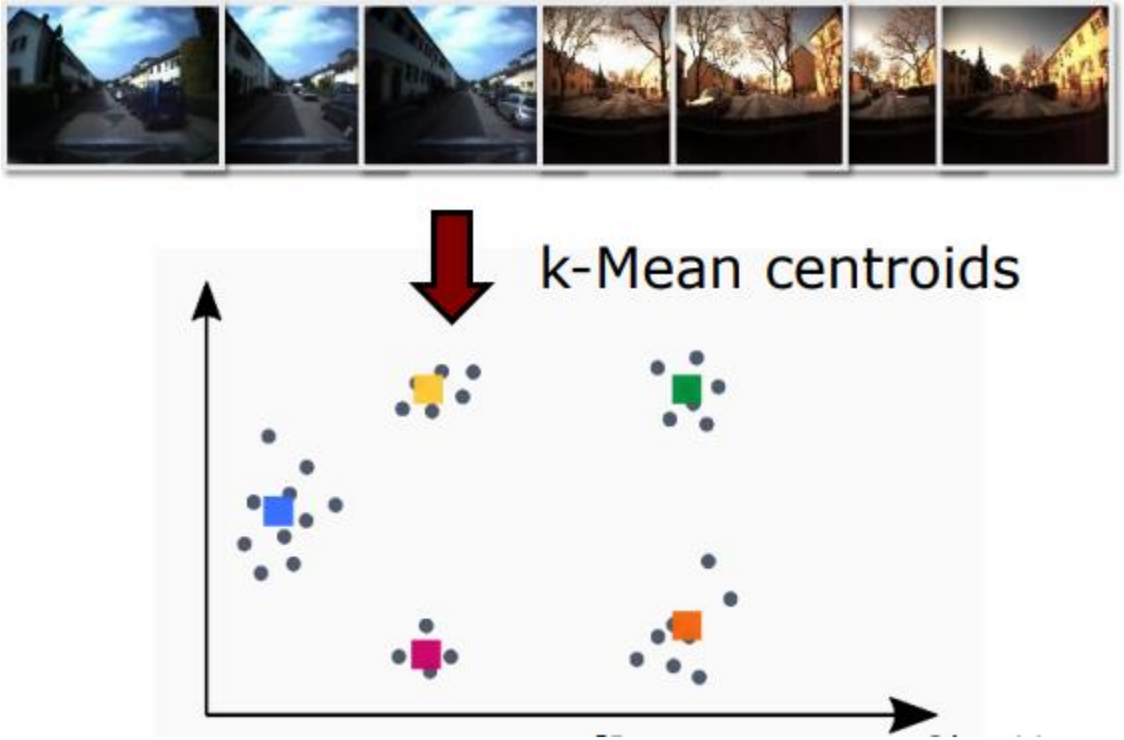
Summary K-Means

- Standard approach to clustering
- Simple to implement
- Number of clusters k must be chosen

- Depends on the initialization
- Sensitive to outliers
- Prone to local minima

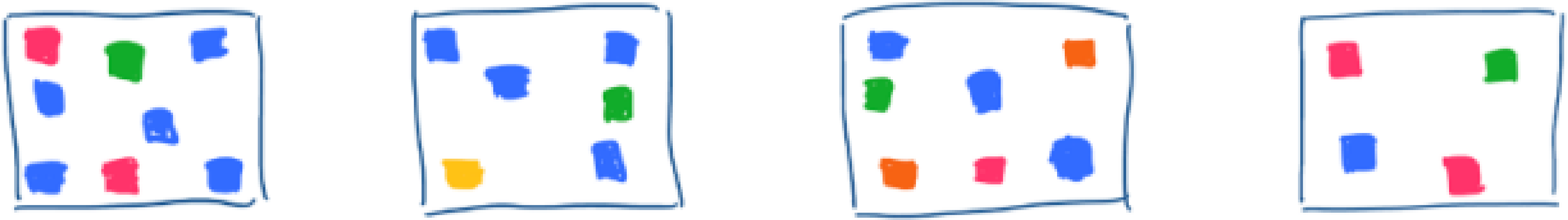
We use k-means to compute the dictionary of visual words

K-Means for Building the Dictionary from Training Data



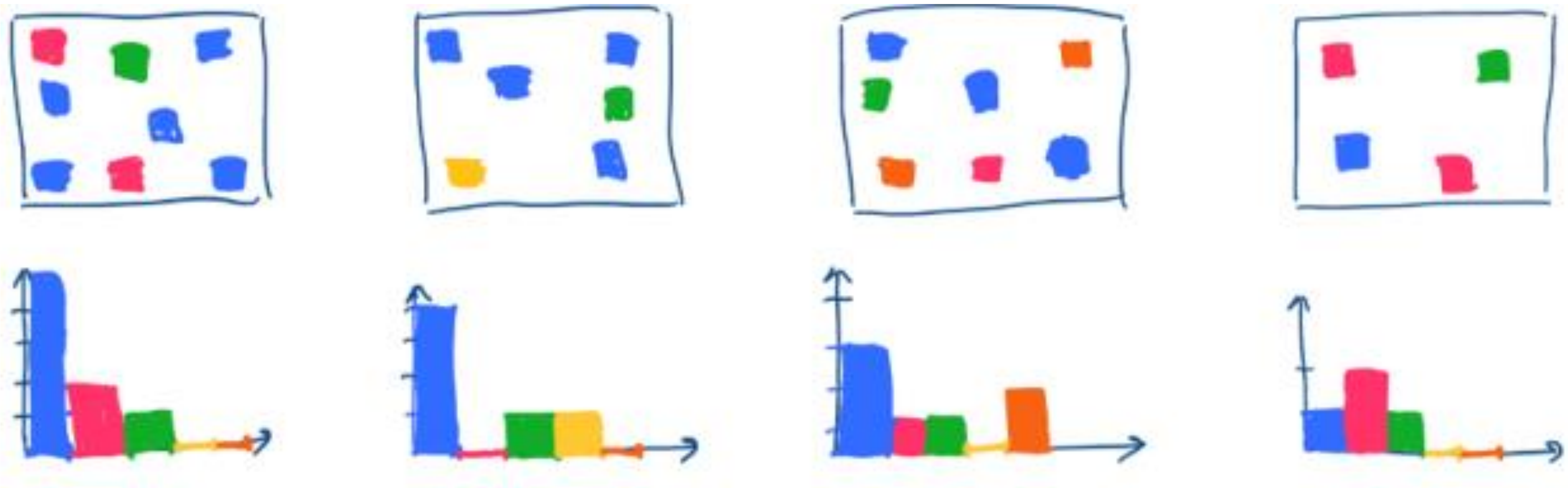
[Image courtesy: Olga Vysotska]

All Images are Reduced to Visual Words



[Image courtesy: Olga Vysotska]

All Images are Represented by Visual Word Occurrences



Every image turns into a histogram

[Image courtesy: Olga Vysotska]

Bag of Visual Words Model

- Compact summary of the image content
- Largely invariant to viewpoint changes and deformations
- Ignores the spatial arrangement

- Unclear how to choose optimal size of the vocabulary
  - Too small: Words not representative of all image regions
  - Too large: Over-fitting

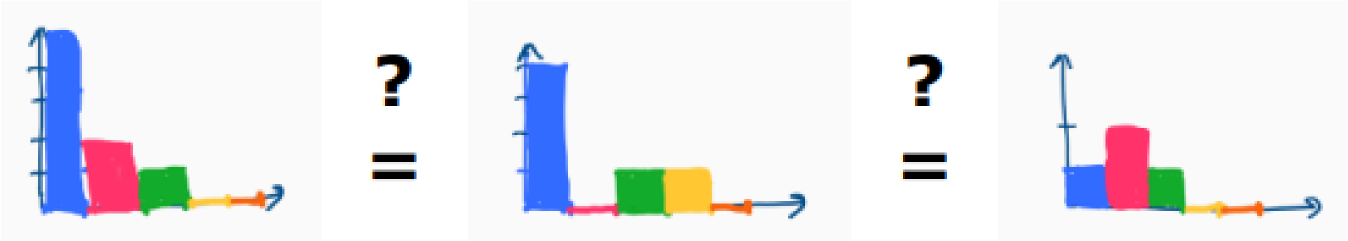
How to Find Similar Images?

Task Description

- Task: Find similar looking images
- Input:
  - Database of images
  - Dictionary
  - Query image(s)

- Output:
  - The N most similar database images to the query image

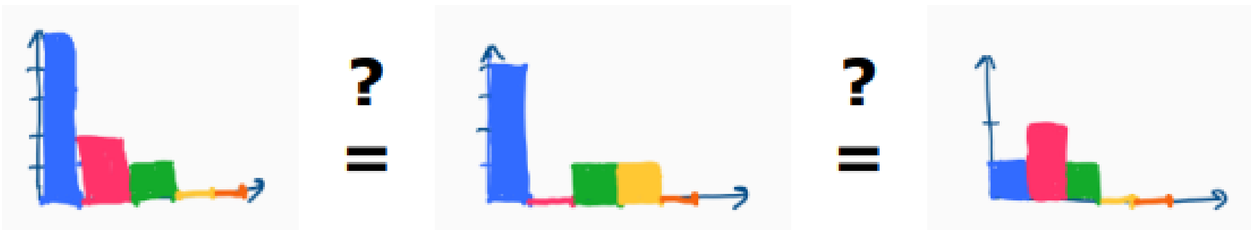
Image Similarity by Comparing Word Occurrence Histograms



[Image courtesy: Olga Vysotska]

How to Compare Histograms?

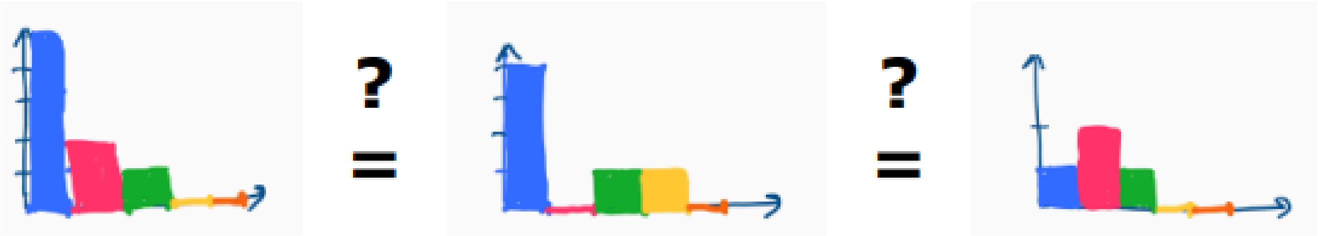
- Euclidean distance of two points?
- Angle between two vectors?
- Kullback Leibler divergence (KLD)?
- Something else?



[Image courtesy: Olga Vysotska]

Are All Words Expressive for Comparing Histograms?

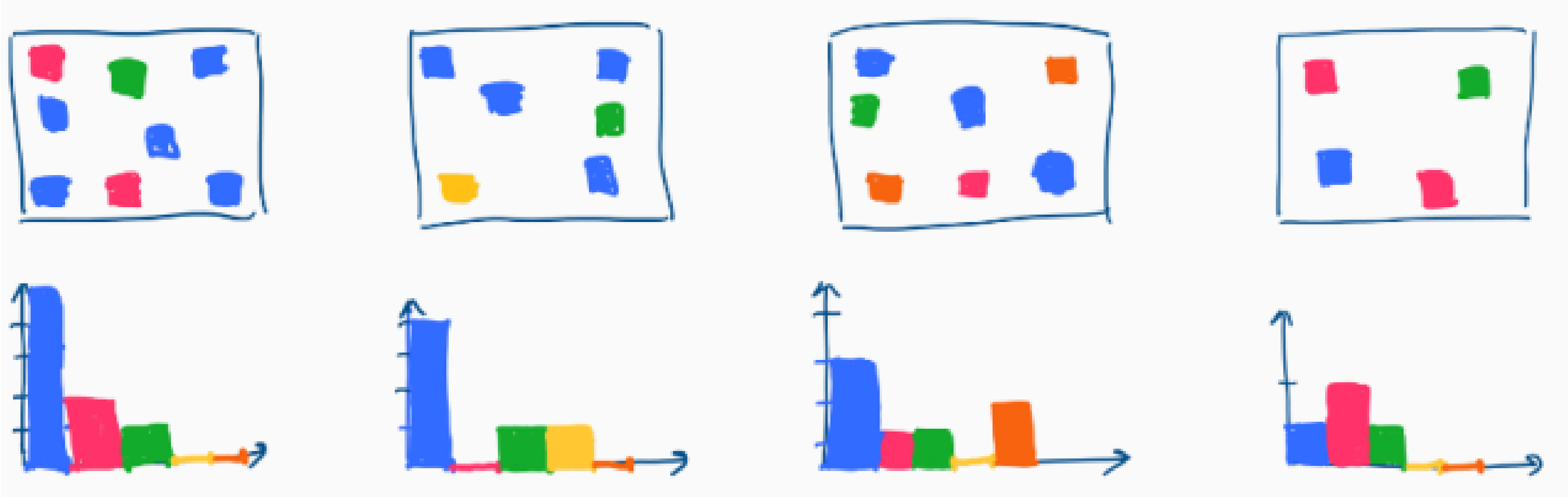
- Should all visual words be treated in the same way?
- Text analogy: What about articles?
  - E.g “the”, “a”, … they can be found in all documents.



[Image courtesy: Olga Vysotska]

Some Word are Less Expressive Than Others!

- Words that occur in every image do not help a lot for comparisons



- Example: the "green word" is useless

[Image courtesy: Olga Vysotska]

TF-IDF Reweighting

- Weight words considering the probability that they appear
- $TF - IDF \; = \; term\ frequency – \; inverse\ document\ frequency$
- Every bin is reweighted

Compute visual word I in image d

$$\underbrace{t_{id}}_{\text{bin}} = \underbrace{\frac{n_{id}}{n_d}}_{\text{normalize}} \underbrace{\log \frac{N}{n_i}}_{\text{weight}}$$

TF-IDF

term frequency

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

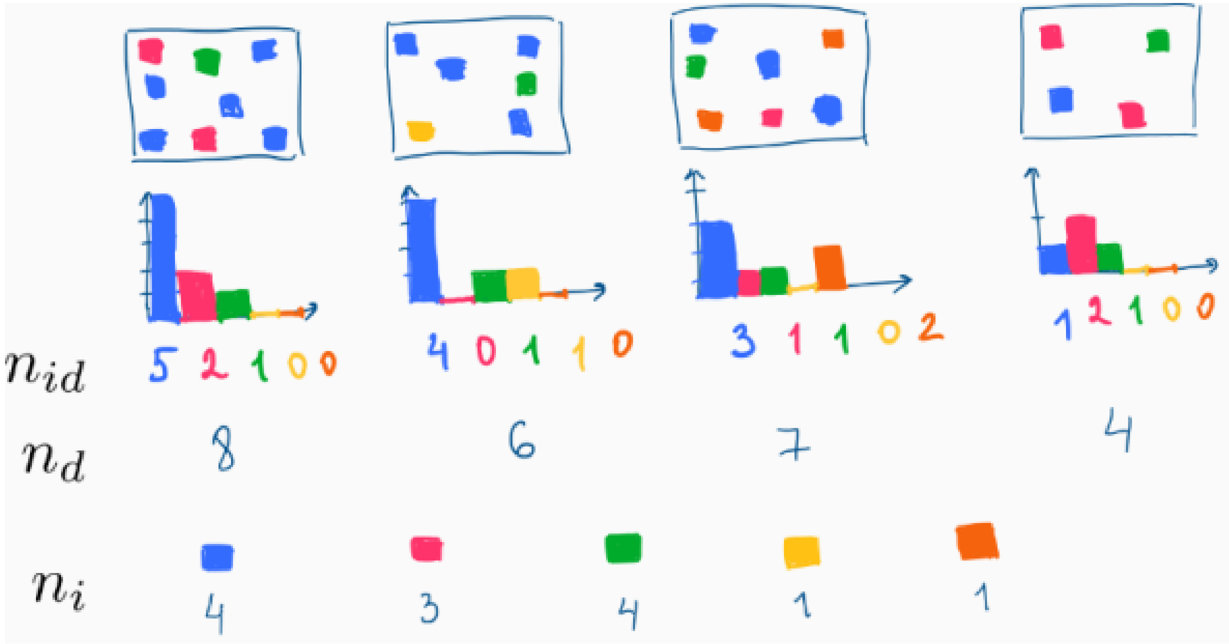
inverse document frequency

bin of word i in image d

- t_{id}: histogram bin of word i for image d
- n_{id}: occurances of word i in image d
- n_d: number of word occurances in image d
- n_i: number of images that contain word i
- N: number of images

- E.g. if word n_i appear in all image N. $\log 1$ is zero. Thus that word is became irrelevant or get weight 0

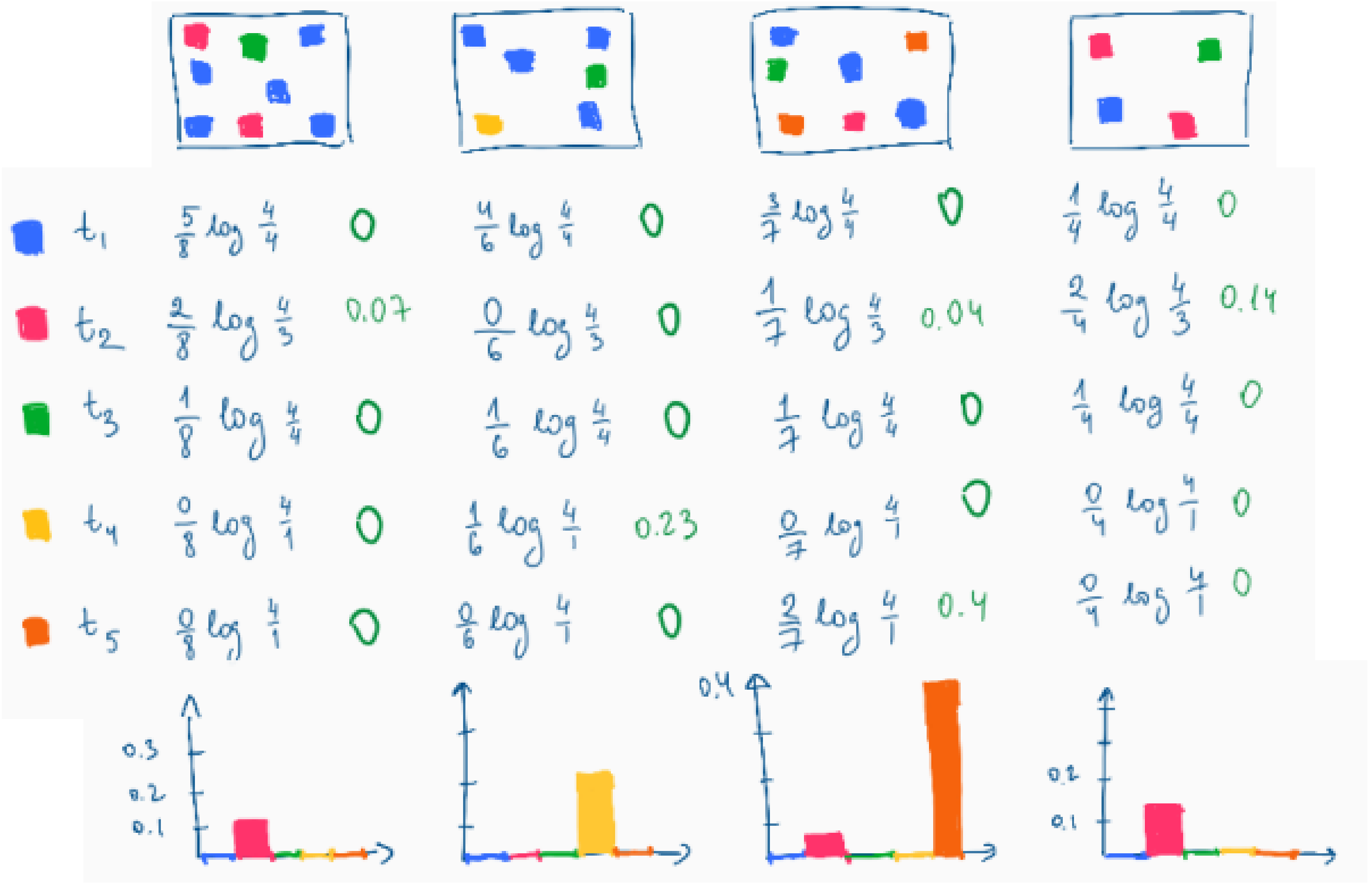
Computing the TF-IDF (1)



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

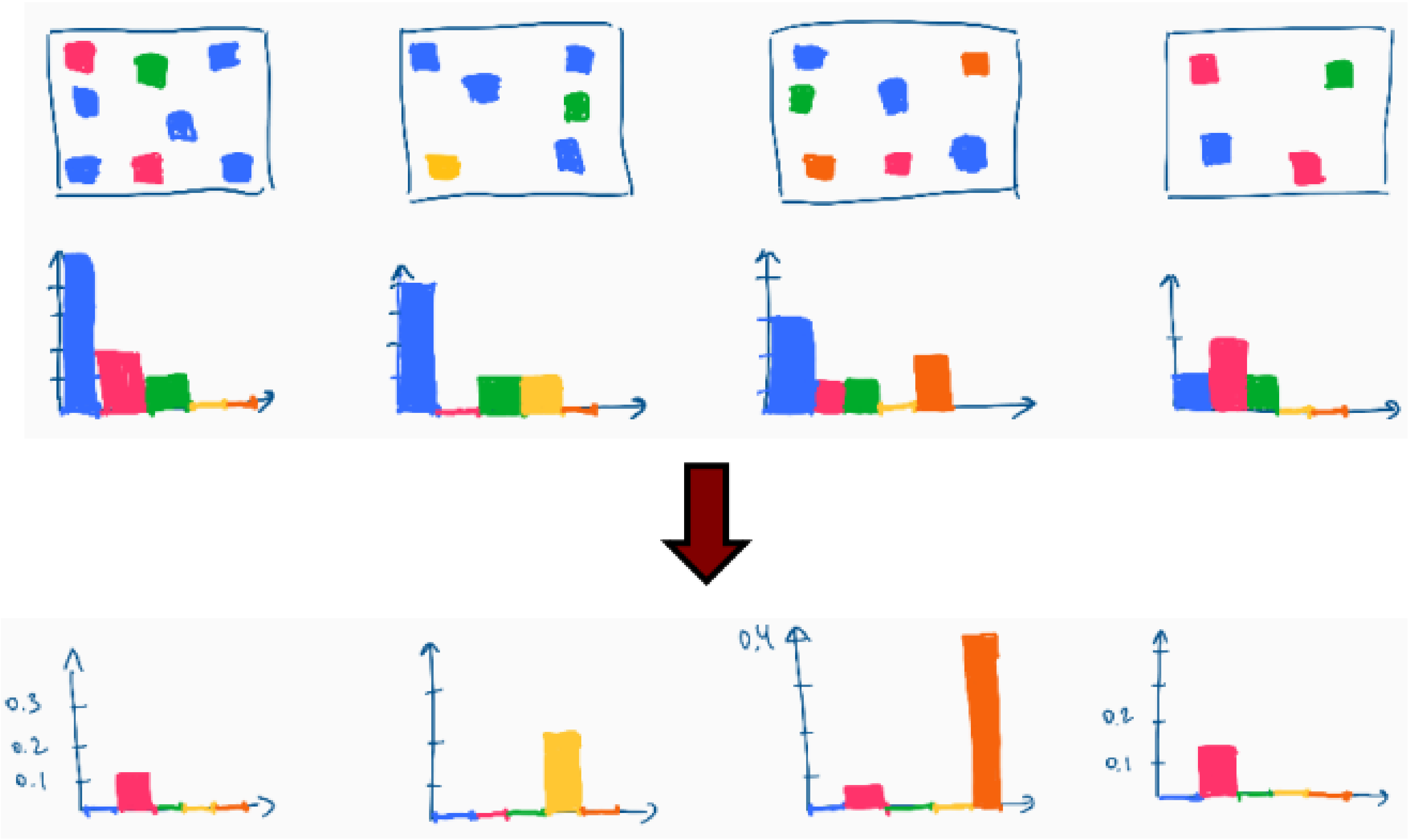
[Image courtesy: Olga Vysotska]

Computing the TF-IDF (2)



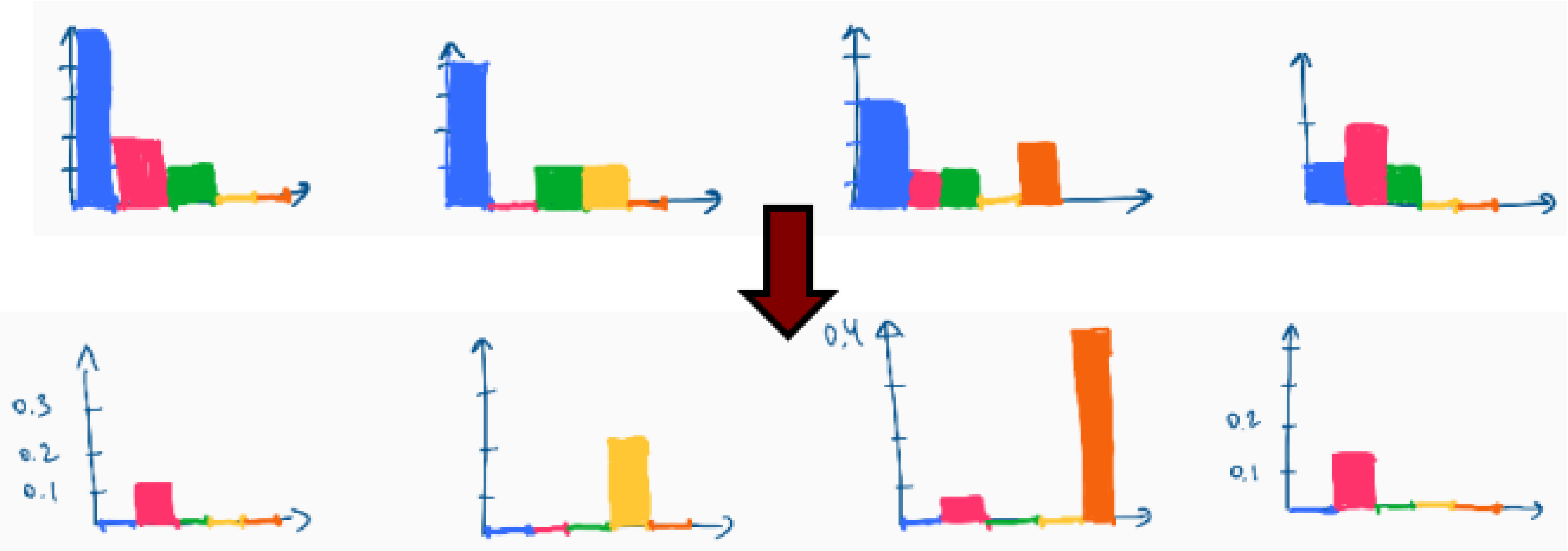
[Image courtesy: Olga Vysotska]

Reweighted Histograms



[Image courtesy: Olga Vysotska]

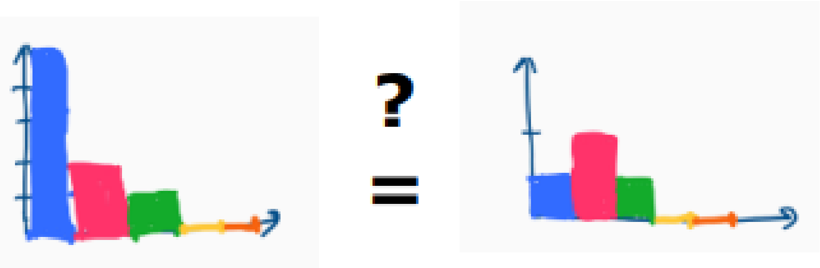
Reweighted Histograms



- Relevant words get higher weights
- Others are weighted down to zero (those occurring in every image)

[Image courtesy: Olga Vysotska]

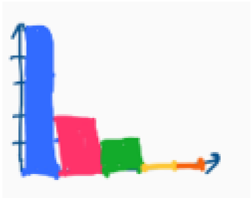
Comparing Two Histograms



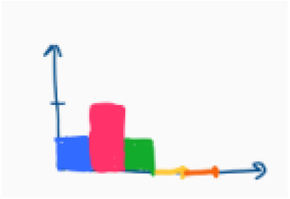
- Options
  - Euclidean distance of two points
  - Angle between two vectors
  - ~~Kullback Leibler divergence (KLD)~~

[Image courtesy: Olga Vysotska]

Comparing Two Histograms



$\stackrel{?}{=}$



- Options
  - Euclidean distance of two vectors
  - Angle between two vectors
  - ~~Kullback Leibler divergence (KLD)~~

BoVW approaches often use the cosine distance for comparisons

Cosine Similarity and Distance

- Cosine similarity considers the cosine of the angle between vectors:

$$\mathrm{cossim}(\mathbf{x}, \mathbf{y}) = \cos(\theta) = \frac{\mathbf{x}^\top \mathbf{y}}{||\mathbf{x}||\,||\mathbf{y}||}$$

- We use the cosine distance

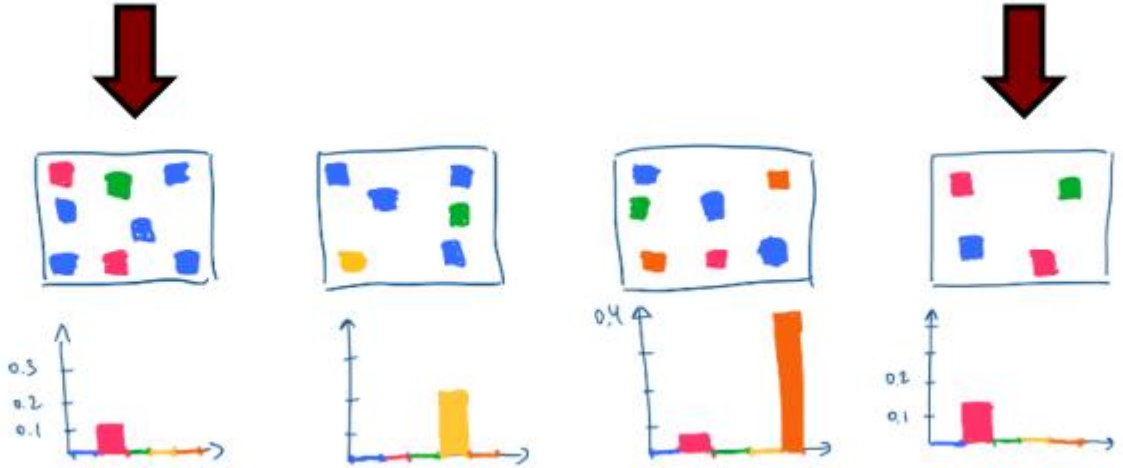
$$\mathrm{d}_{\cos}(\mathbf{x}, \mathbf{y}) = 1 - \mathrm{cossim}(\mathbf{x}, \mathbf{y}) = 1 - \frac{\mathbf{x}^\top \mathbf{y}}{||\mathbf{x}||\,||\mathbf{y}||}$$

- Takes values between 0 and 1 (for vectors in the 1st quadrant)

- Close to 0 zero means the image is similar and close to 1 means the image is not similar.

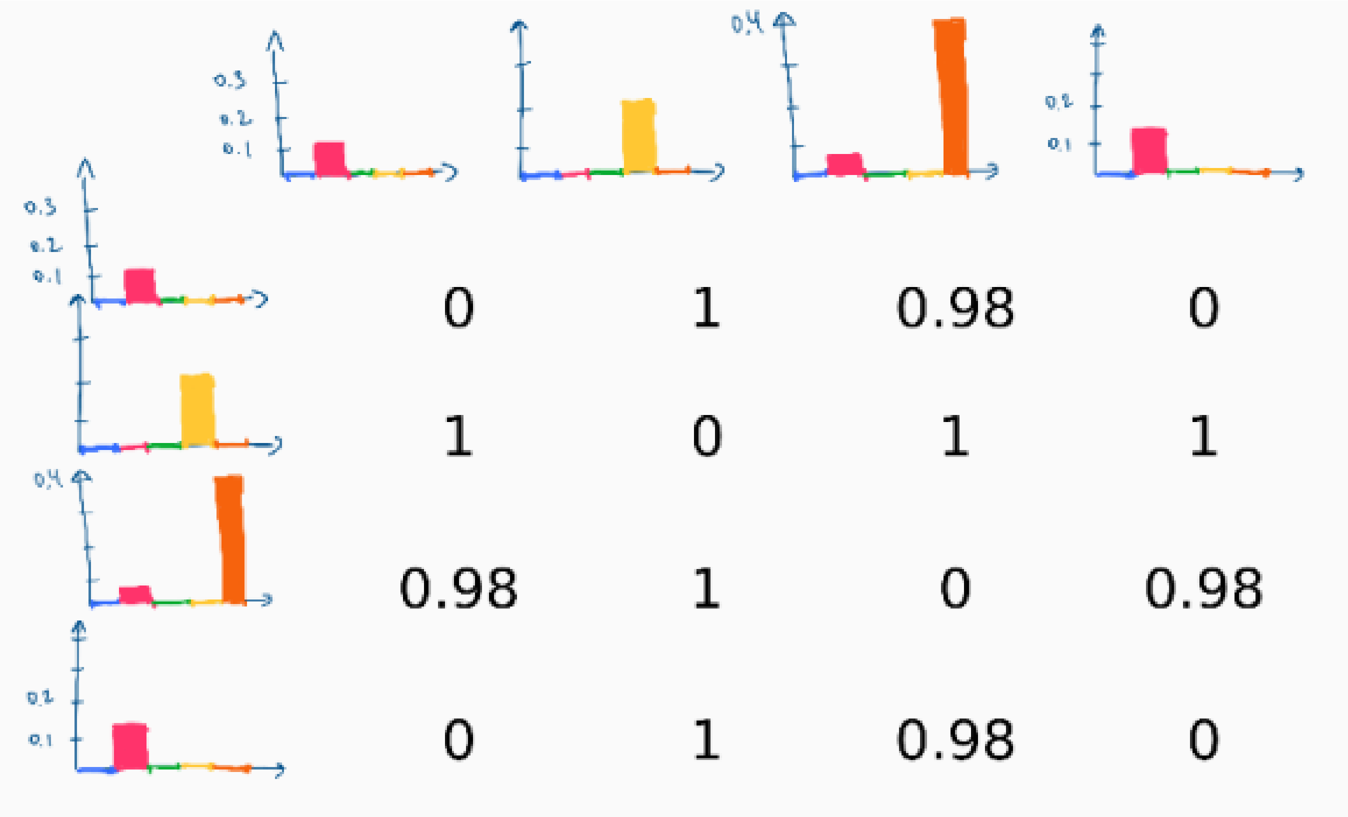
Example Comparing Histograms

- 4 images
- Image 0 and image 3 are similar



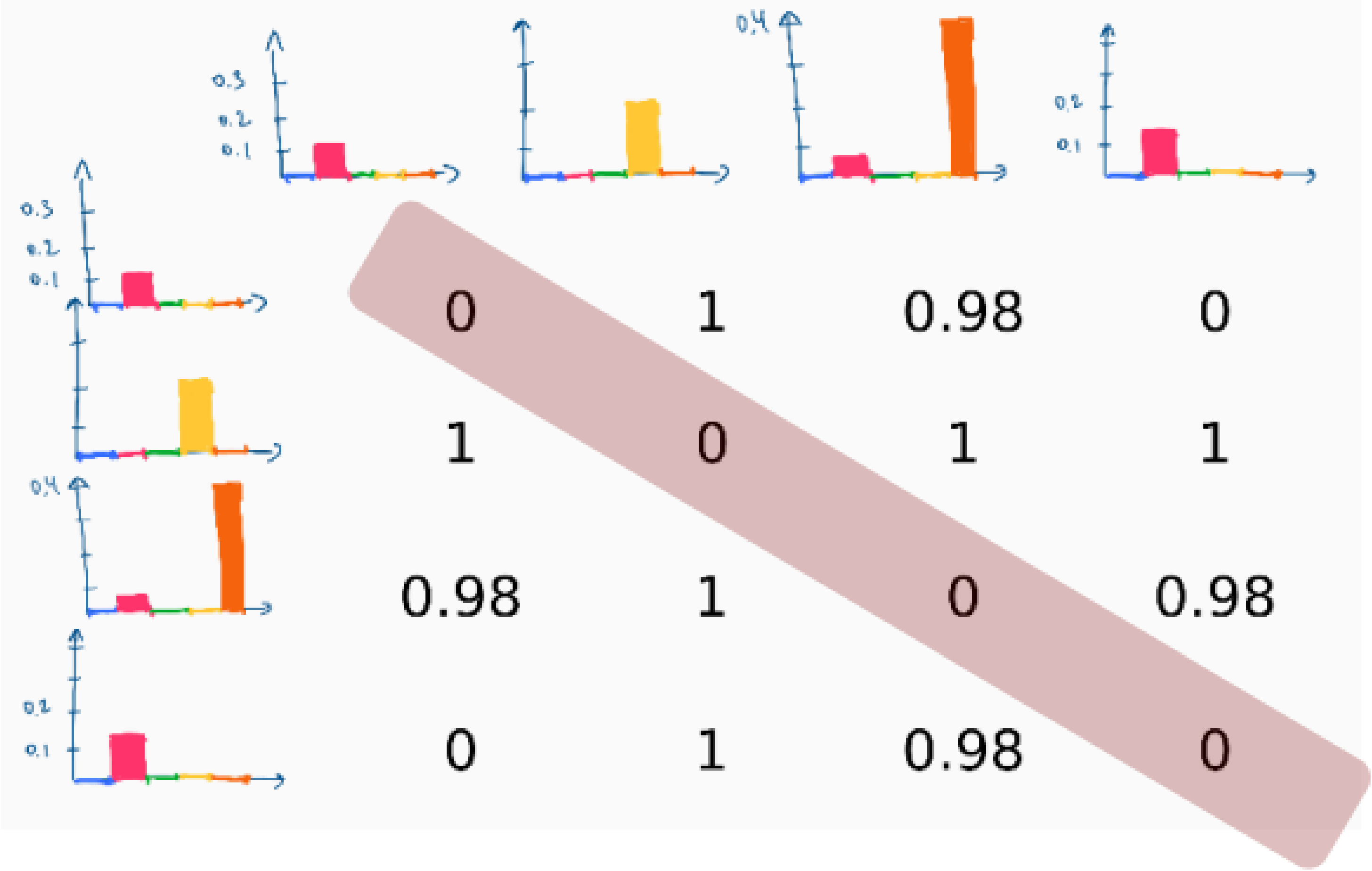
[Image courtesy: Olga Vysotska]

Example Comparing Histograms



[Image courtesy: Olga Vysotska]

Example Comparing Histograms

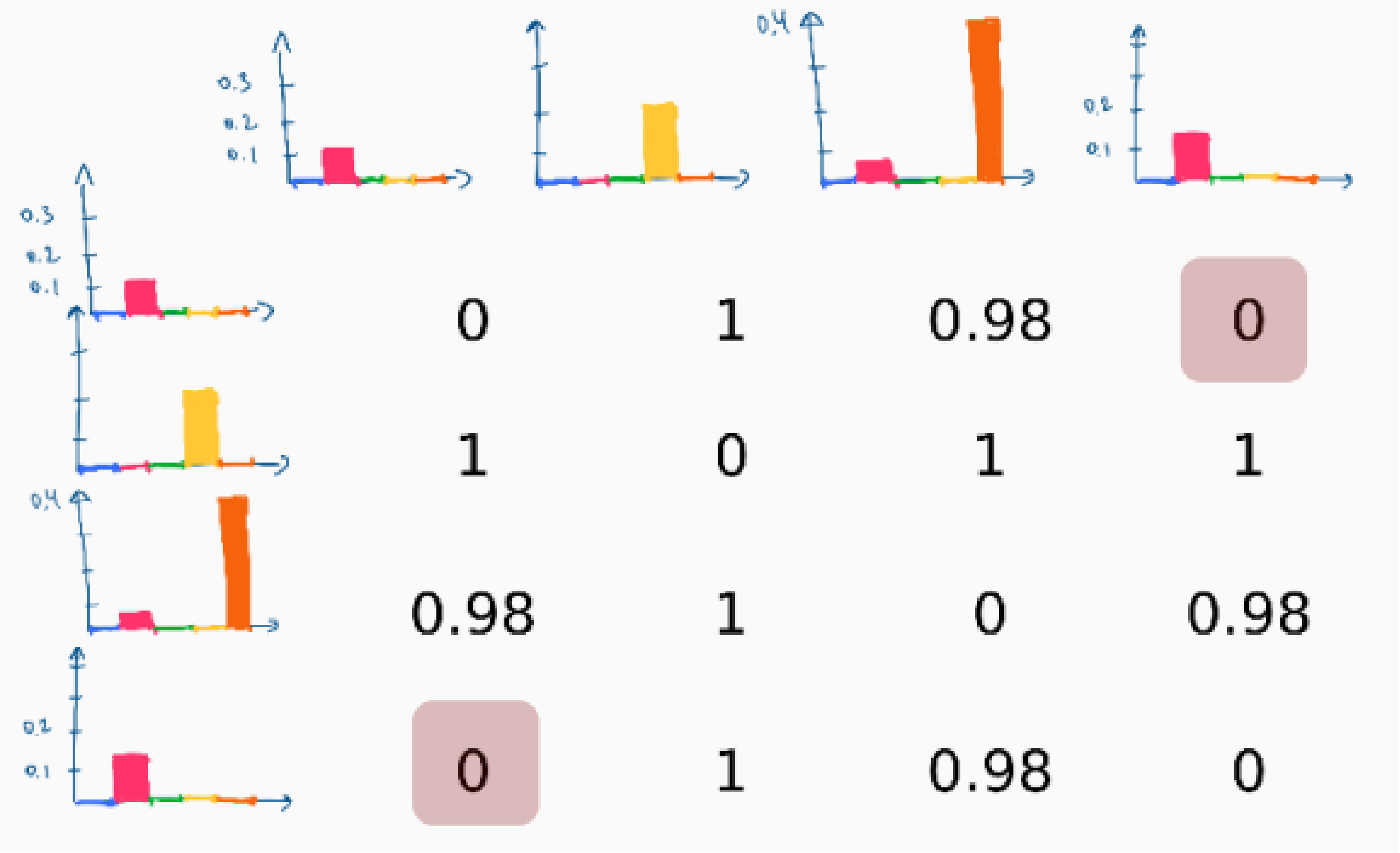


0	1	0.98	0
1	0	1	1
0.98	1	0	0.98
0	1	0.98	0

Images have a zero distance to themselves

[Image courtesy: Olga Vysotska]

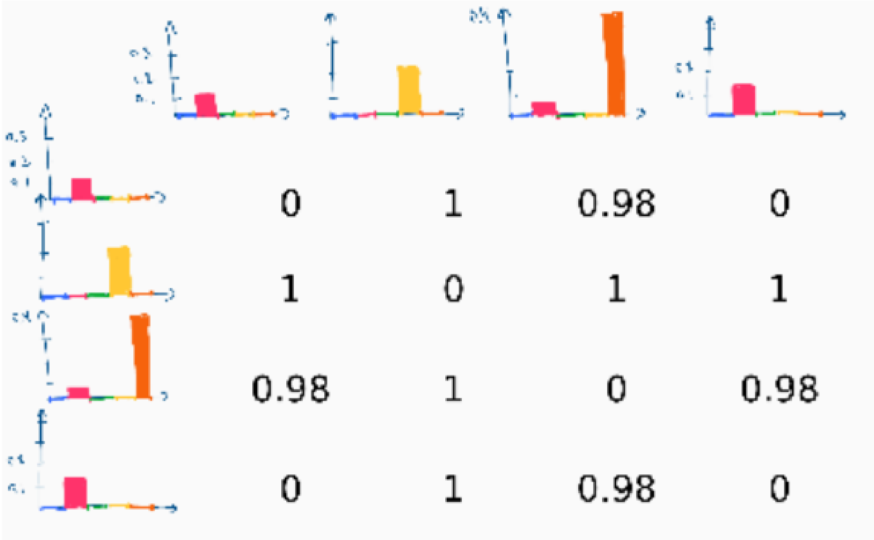
Example Comparing Histograms

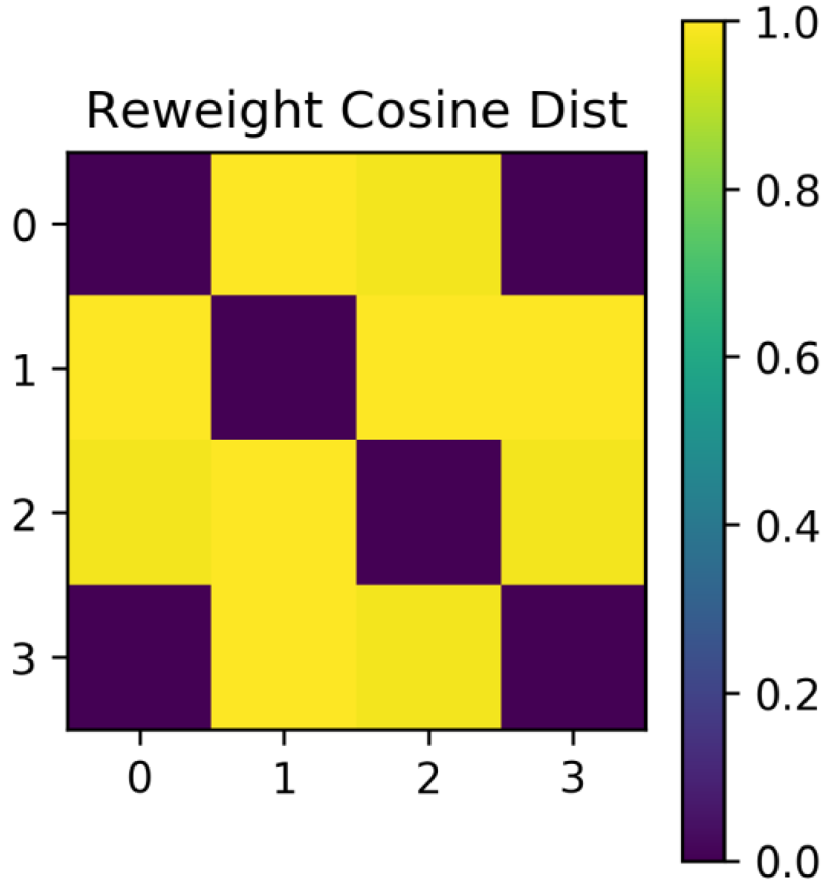


Images 0 and 3 are highly similar

[Image courtesy: Olga Vysotska]

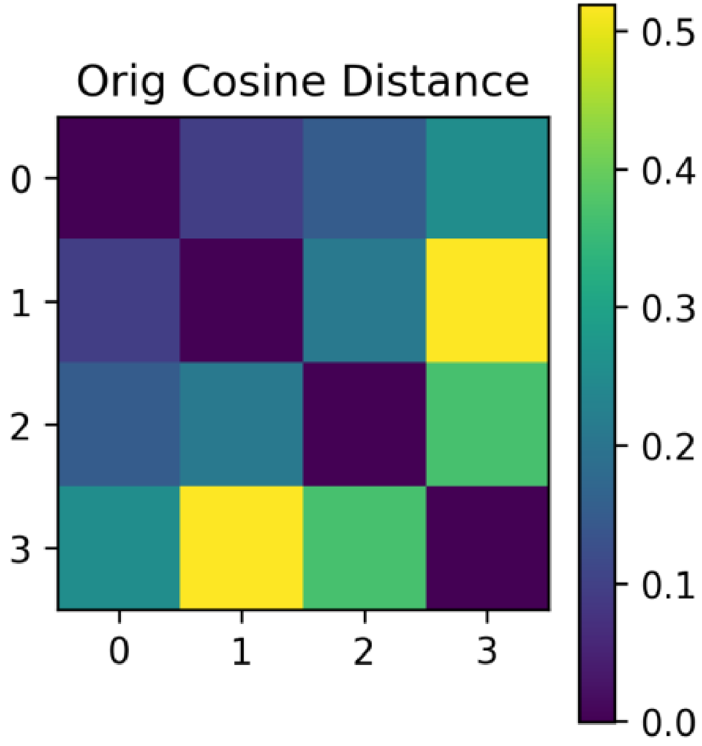
Cost Matrix

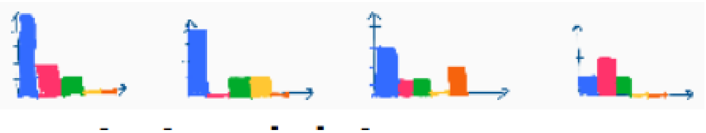




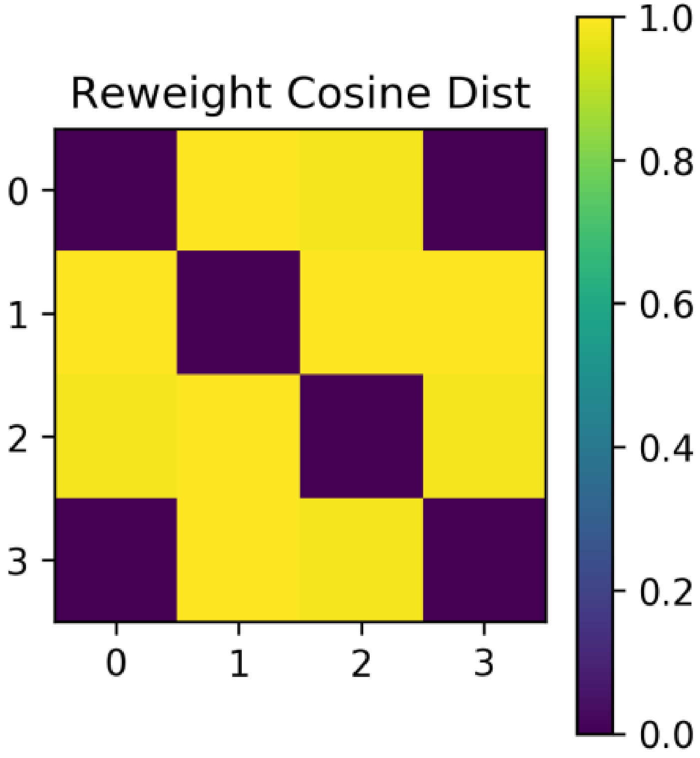
[Image courtesy: Olga Vysotska]

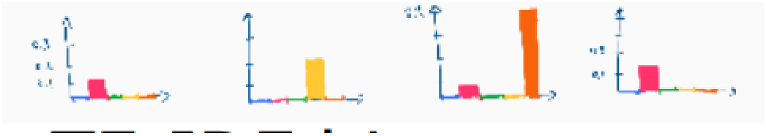
IF-IDF Actually Helps





Original histograms

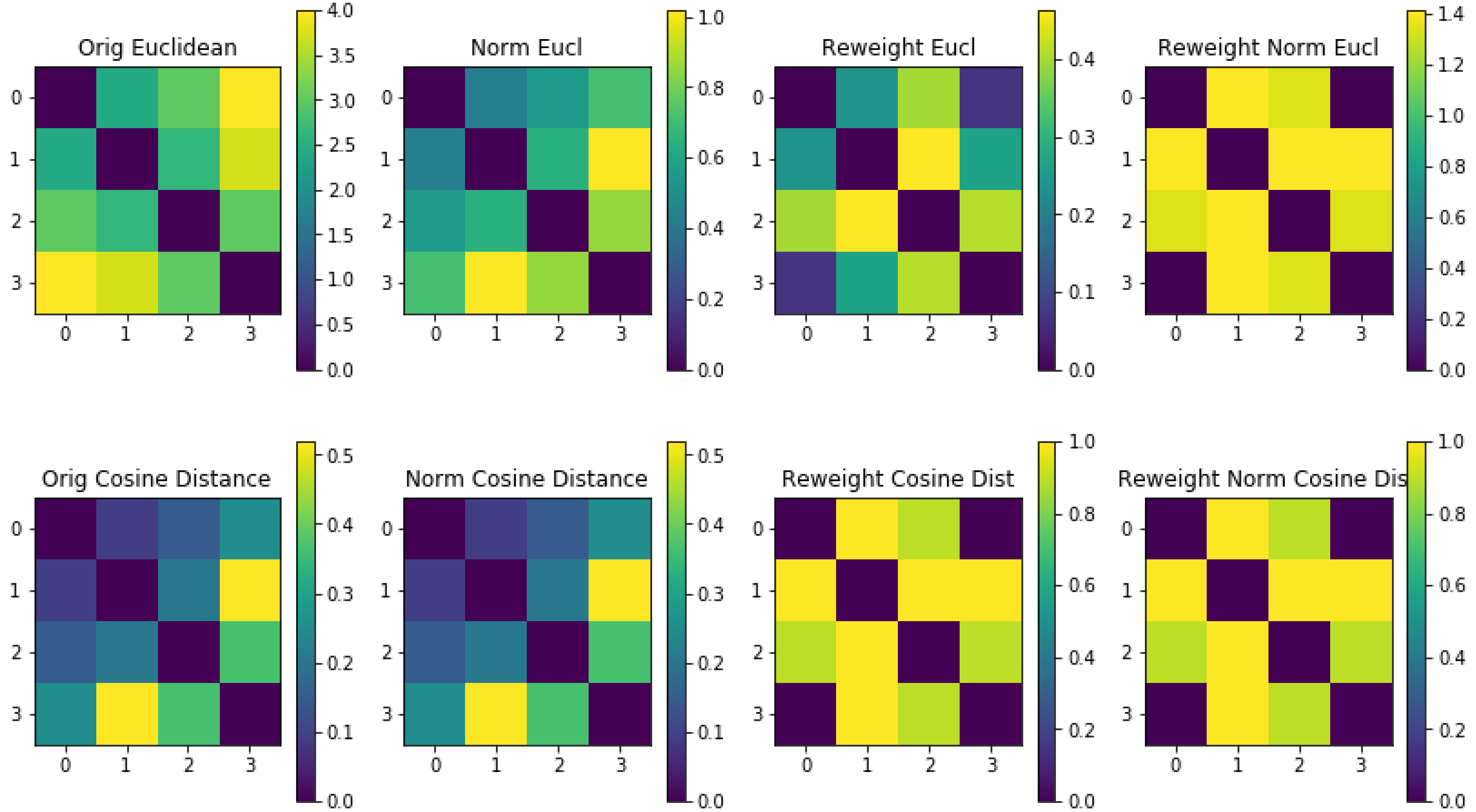




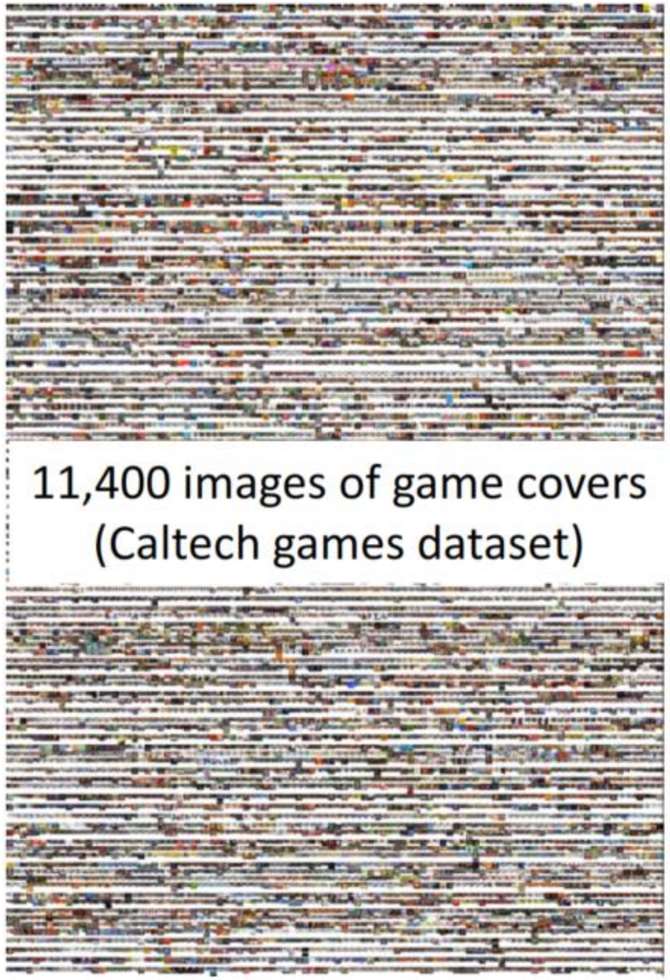
TF-IDF histograms

[Image courtesy: Olga Vysotska]

Euclidean VS Cosine distance



Large-scale image matching



- Bag-of-words models have been useful in matching an image to a large database of object instances.

How do I find this image in the database?

[Image courtesy: Fei-Fei Li]

Large-scale image search

- Build the database:
  - Extract features from the database images
  - Learn a vocabulary using k-means (typical k: 100,000)
  - Compute weights for each word
  - Create an inverted file mapping words →images

[Image courtesy: Fei-Fei Li]

Similarity Queries

- Database stores TF-IDF weighted histograms for all database images
- Find similar images by
  - Extract features from query image
  - Assign features to visual words
  - Build TF-IDF histogram for query image
  - Return N most similar histograms from database under cosine distance

Large-scale image search

- Cons:
  - Performance degrades as the database grows

[Image courtesy: Fei-Fei Li]

Large-scale image search

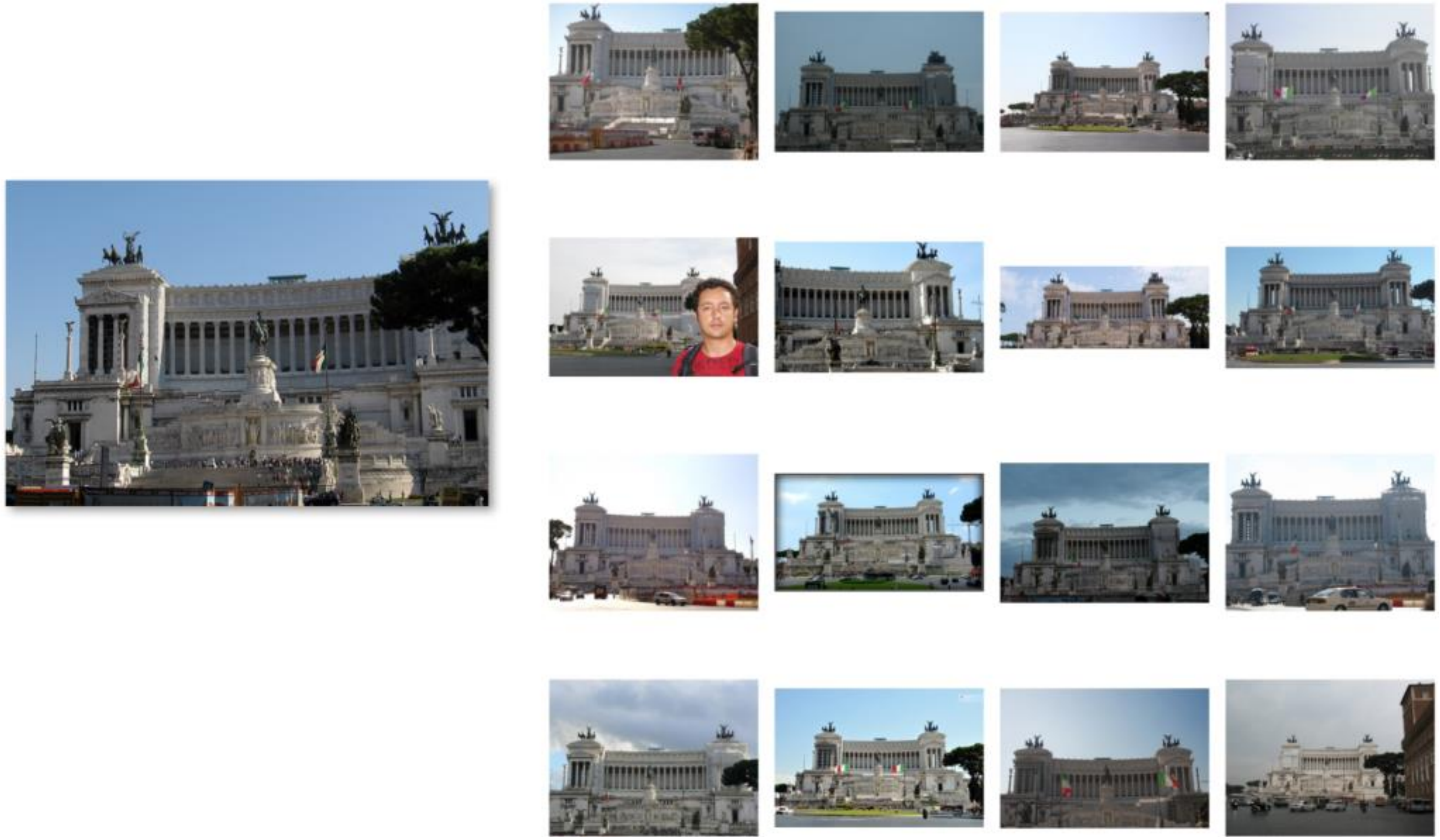
- Pros:
  - Works well for CD covers, movie posters
  - Real-time performance possible

real-time retrieval from a database of 40,000 CD covers

Nister & Stewenius,**Scalable Recognition with a Vocabulary Tree**

[Image courtesy: Fei-Fei Li]

Example bag-of-words matches



[Image courtesy: Fei-Fei Li]

Example bag-of-words matches

[Image courtesy: Fei-Fei Li]

Recap

- BoVW is an approach to compactly describe images and compute similarities between images
- Based in a set of visual words
- Images become histograms of visual word occurrences
- TF-IDF weighting for increasing the influence of expressive words
- Similarity = histogram similarity
- Cosine distance

References/Further Material

- Jupyter notebook by Olga Vysotska:
  - https://github.com/ovysotska/in_simple_english/blob/master/bag_of_visual_words.ipynb
- Sivic and Zisserman. Video Google:
  - A Text Retrieval Approach to Object Matching in Videos, 2003:
  - http://www.robots.ox.ac.uk/~vgg/publications/papers/sivic03.pdf
- TF-IDF information:
  - https://en.wikipedia.org/wiki/Tf%E2%80%93idf

Next: Viola-Jones Object Detection