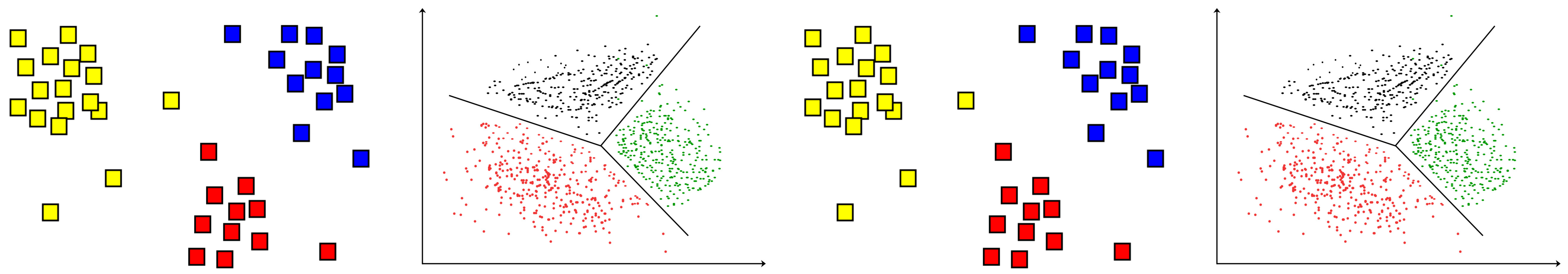
THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision



Lecture 19: Image Classification (cont.)

Menu for Today (**March 19, 2020**)

Topics:

— Scene Classification

— Bag of Words Representation

— Decision Tree

— Boosting

Redings:

— **Today's** Lecture: Forsyth & Ponce (2nd ed.) 16.1.3, 16.1.4, 16.1.9

— **Next** Lecture: Forsyth & Ponce (2nd ed.) 17.1–17.2

Reminders:

— No more **fun** examples (sorry) … it is difficult to get them to work

— **Assignment 5** is out

Lecture 18: Re-cap

Classify images containing single **objects**, the same techniques can be applied to classify natural **scenes** (e.g. beach, forest, harbour, library).

Lecture 18: Re-cap (Vector Space Model)

Many algorithms for image classification accumulate evidence on the basis of **visual words**.

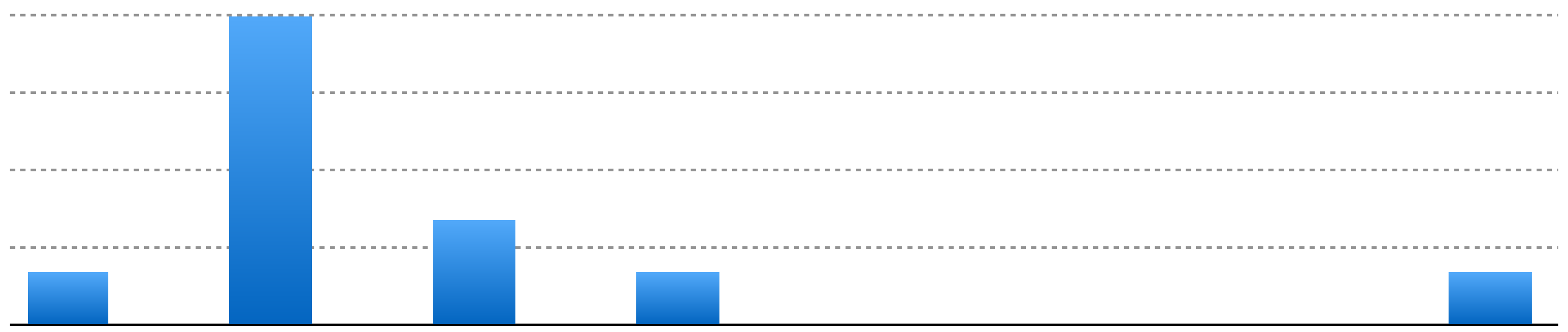
To classify a text document (e.g. as an article on sports, entertainment, business, politics) we might find patterns in the occurrences of certain words.

Lecture 18: Re-cap (Vector Space Model)

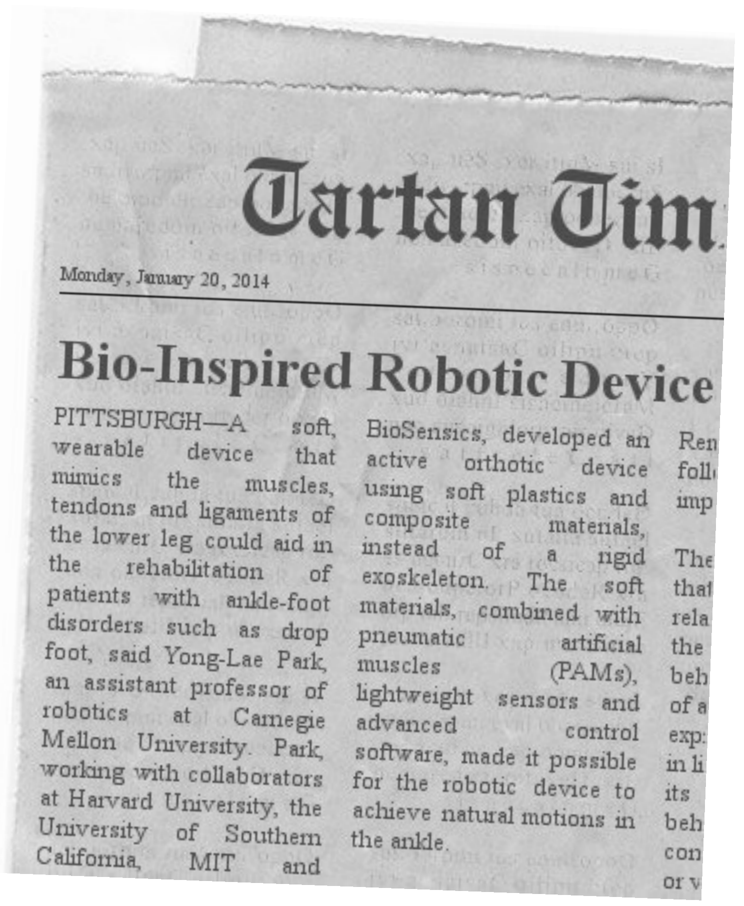
G. Salton. 'Mathematics and Information Retrieval' Journal of Documentation, 1979


The Newspa
Sunday, December 22, 2013
DARPA Selects Carnegie Me

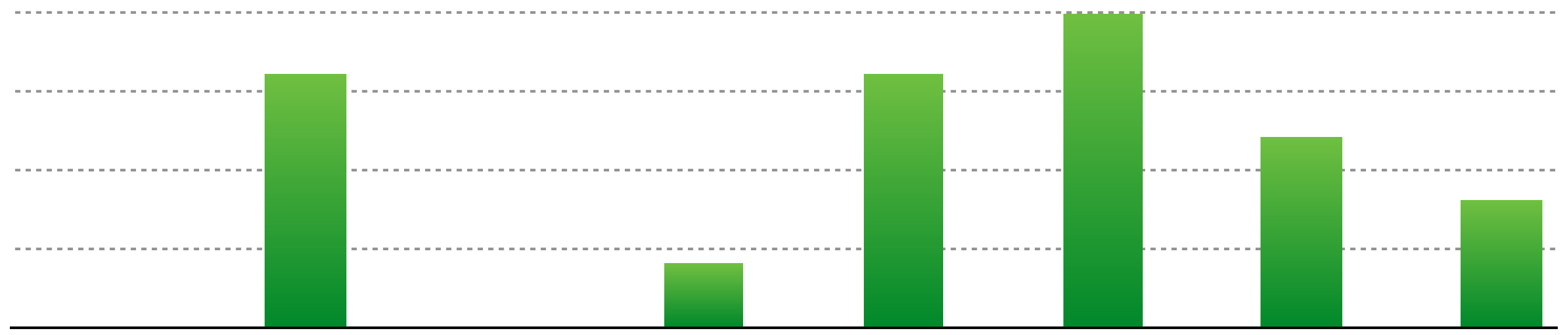
The Tartan Rescue Team from Carnegie Mellon University's National Robotics Engineering Center ranked third among teams competing in the Defense Advanced Research Projects Agency (DARPA) Robotics Challenge Trials this weekend in Homestead, Fla., and was selected by the agency as one of eight teams eligible for DARPA funding to prepare for next December's finals. The team's four-limbed CMU Highly Intelligent Mobile Platform, or CHIMP, robot scored 18 out of a possible 32 points during the two-day trials. It demonstrated its ability to perform such tasks as removing debris, cutting a hole through a wall and closing a series of valves.



1	6	2	1	0	0	0	1
Tartan	robot	CHIMP	CMU	bio	soft	ankle	sensor


Tartan Tim
Monday, January 20, 2014
Bio-Inspired Robotic Device

PITTSBURGH—A soft, wearable device that mimics the muscles, tendons and ligaments of the lower leg could aid in the rehabilitation of patients with ankle-foot disorders such as drop foot, said Yong-Lae Park, an assistant professor of robotics at Carnegie Mellon University. Park, working with collaborators at Harvard University, the University of Southern California, MIT and BioSensics, developed an active orthotic device using soft plastics and composite materials, instead of a rigid exoskeleton. The soft materials, combined with pneumatic artificial muscles (PAMs), lightweight sensors and advanced control software, made it possible for the robotic device to achieve natural motions in the ankle.



0	4	0	1	4	5	3	2
Tartan	robot	CHIMP	CMU	bio	soft	ankle	sensor

http://www.fodey.com/generators/newspaper/snippet.asp

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 18: Re-cap (Vector Space Model)

A document (datapoint) is a vector of counts over each word (feature)

$$\boldsymbol{v}_d = [n(w_{1,d}) \;\; n(w_{2,d}) \;\; \cdots \;\; n(w_{T,d})]$$

$n(\cdot)$ counts the number of occurrences

just a histogram over words

What is the similarity between two documents?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 18: Re-cap (Vector Space Model)

A document (datapoint) is a vector of counts over each word (feature)

$$\boldsymbol{v}_d = [n(w_{1,d}) \;\; n(w_{2,d}) \;\; \cdots \;\; n(w_{T,d})]$$

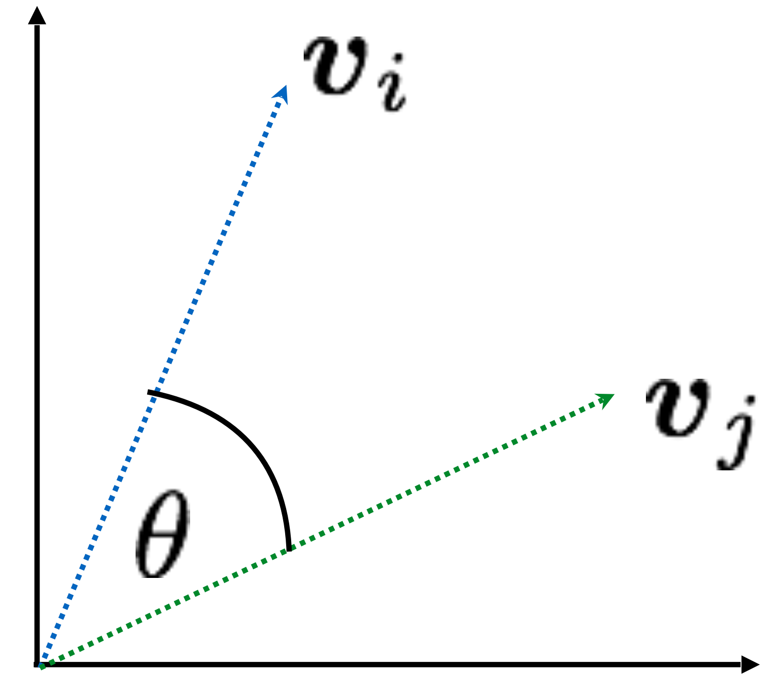
$n(\cdot)$ counts the number of occurrences

just a histogram over words

What is the similarity between two documents?

Use any distance you want but the cosine distance is fast and well designed for high-dimensional vector spaces:

$$d(\boldsymbol{v}_i, \boldsymbol{v}_j) = \cos\theta = \frac{\boldsymbol{v}_i \cdot \boldsymbol{v}_j}{\|\boldsymbol{v}_i\|\|\boldsymbol{v}_j\|}$$



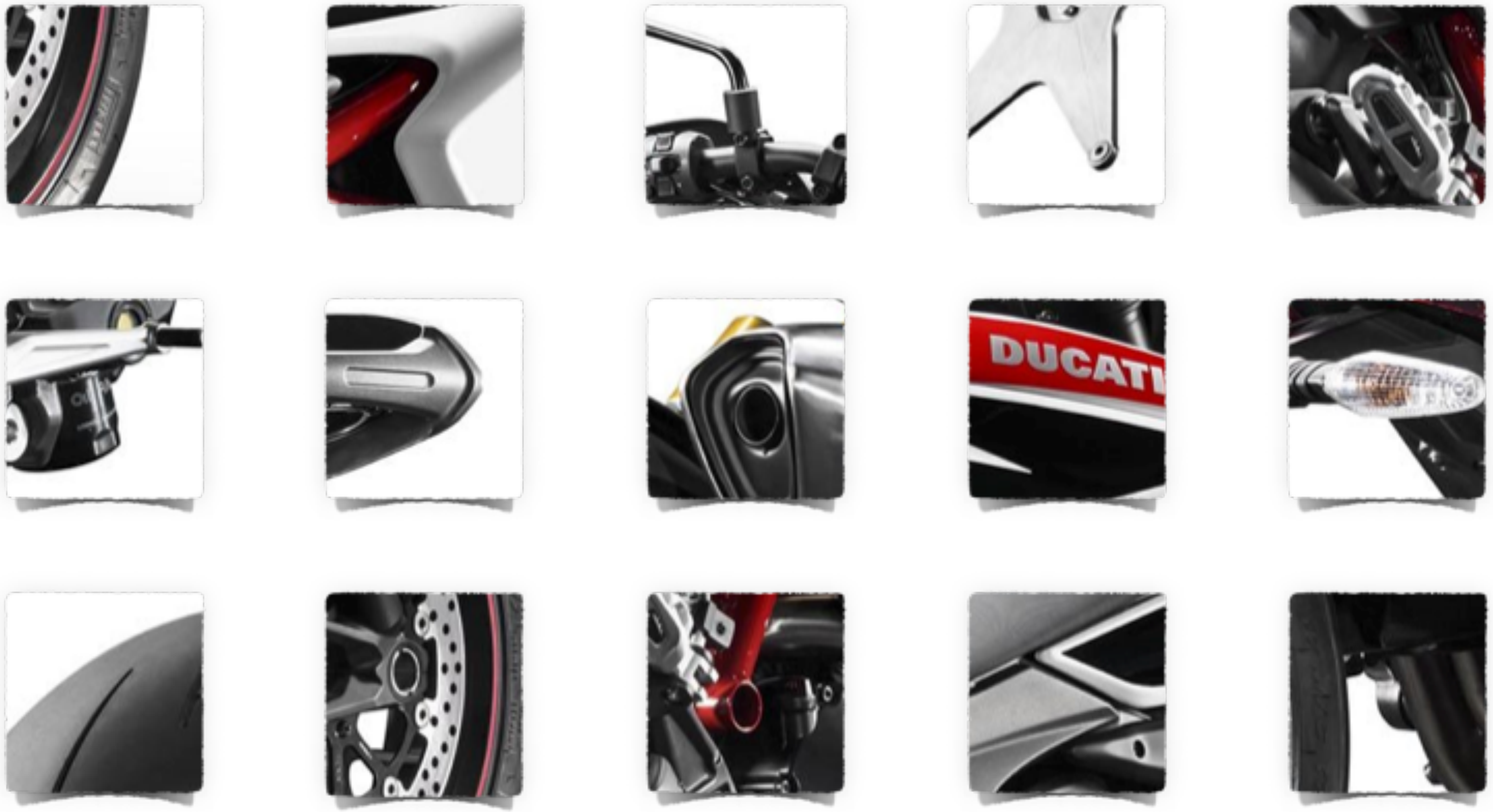
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Visual **Words**

In images, the equivalent of a word is a local image patch. The local image patch is described using a descriptor such as SIFT.

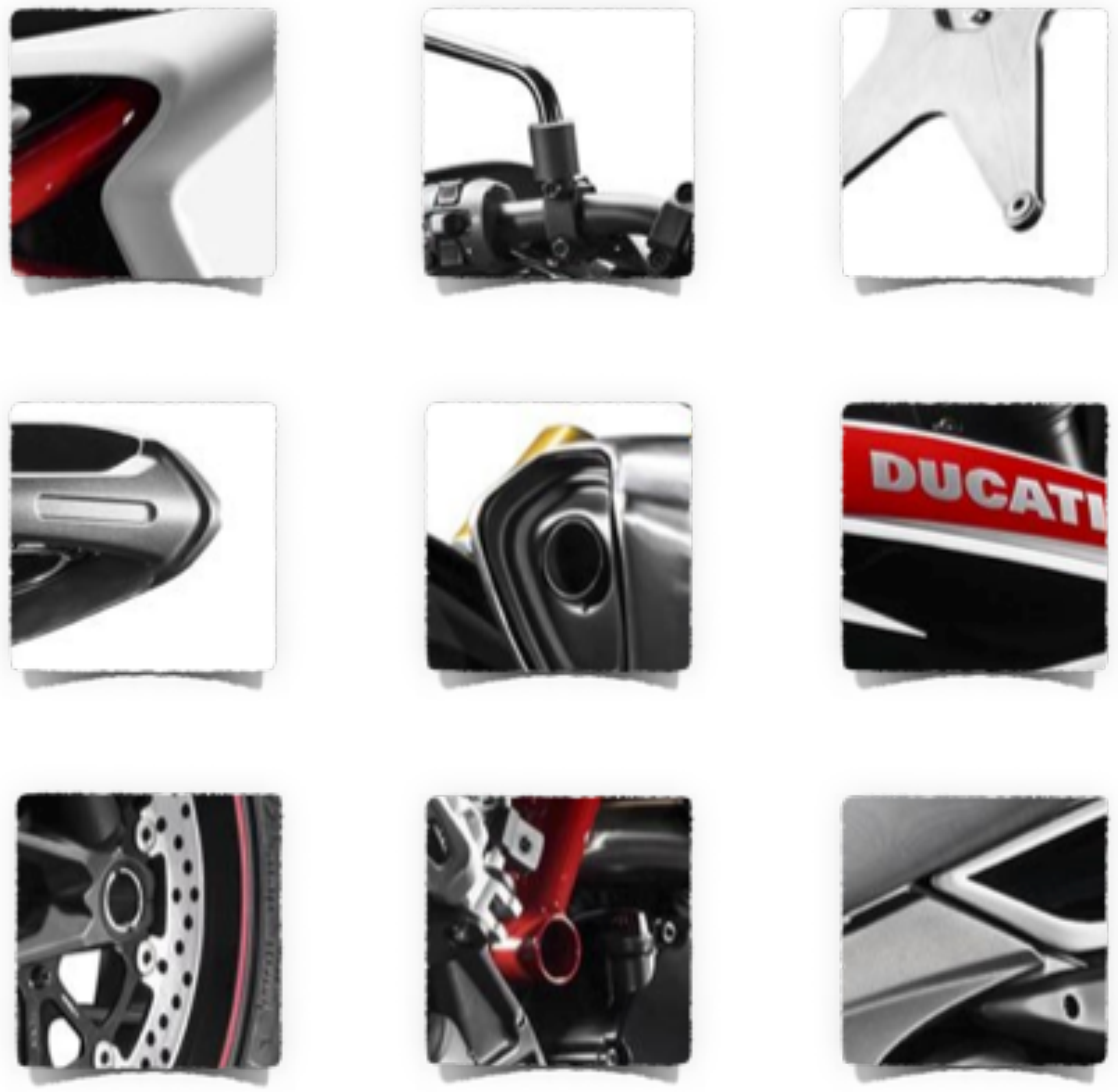
We construct a **vocabulary** or **codebook** of local descriptors, containing representative local descriptors.

What **Objects** do These Parts Belong To?



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Some local feature are very informative



An object as

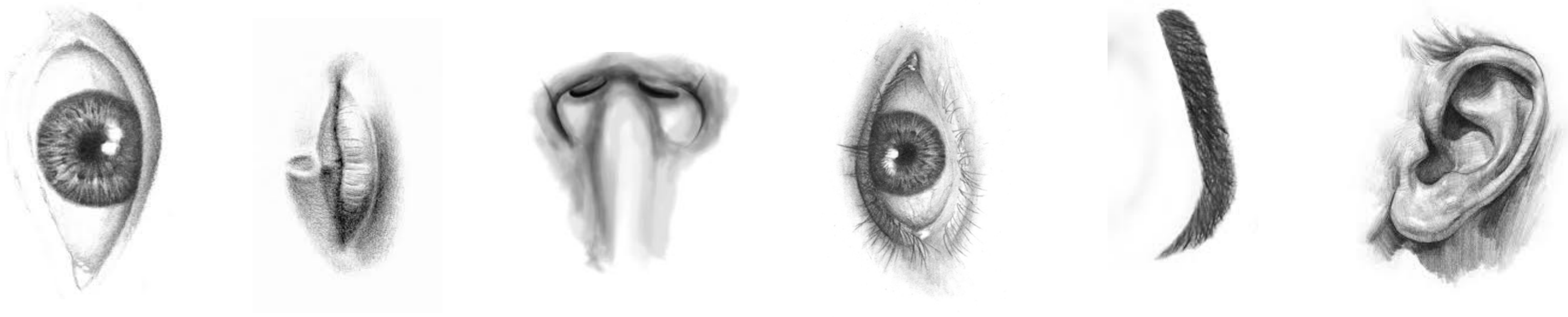
a collection of local features

(bag-of-features)

- deals well with occlusion
- scale invariant
- rotation invariant

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

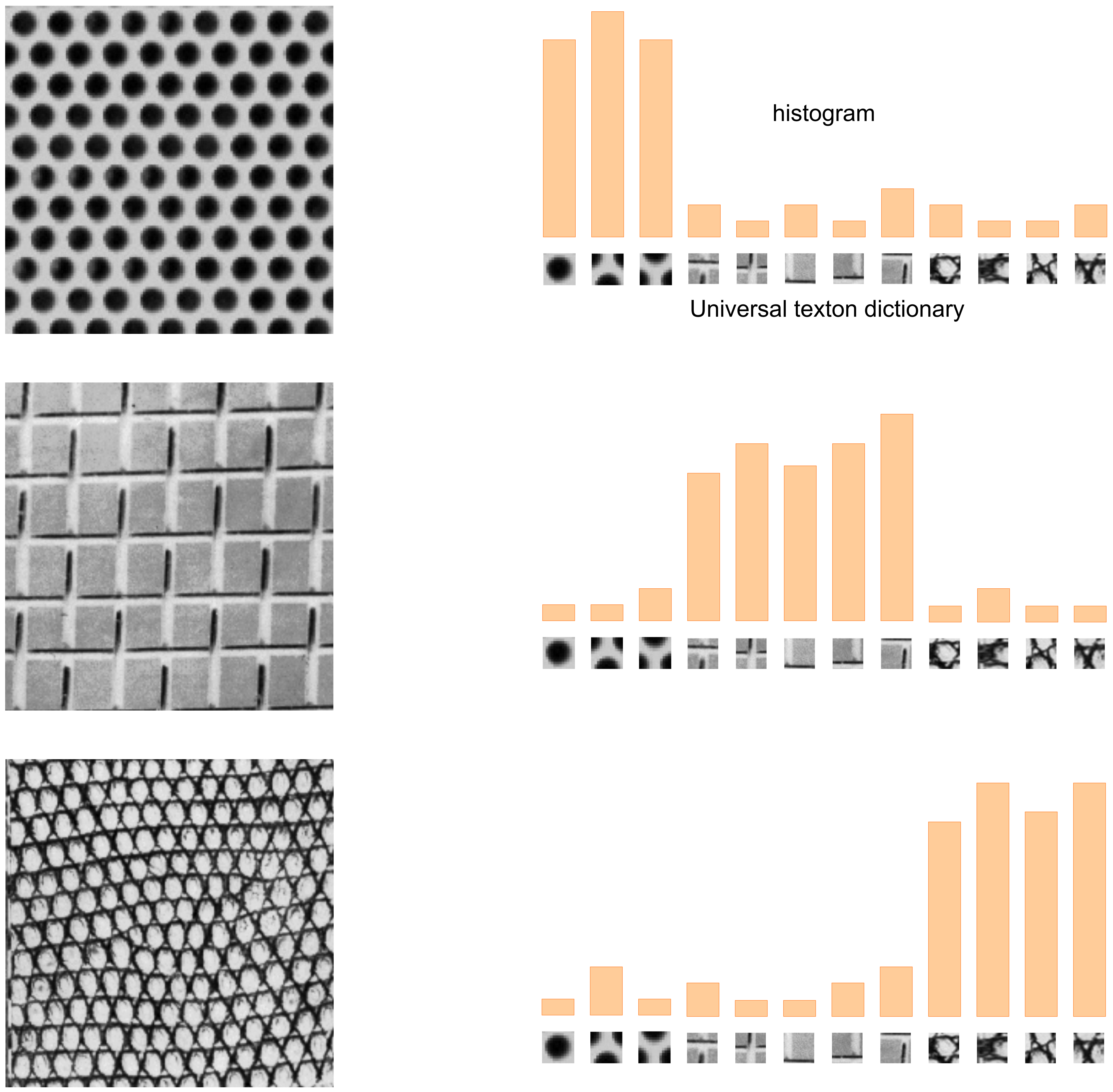
(**not so**) Crazy Assumption



spatial information of local features
can be ignored for object recognition (i.e., verification)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Recall: Texture Representation



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Visual **Words**

In images, the equivalent of a word is a local image patch. The local image patch is described using a descriptor such as SIFT.

We construct a **vocabulary** or **codebook** of local descriptors, containing representative local descriptors.

Question: How might we construct such a codebook? Given a large sample of SIFT descriptors, say 1 million, how can we choose a small number of 'representative' SIFT codewords, say 1000?

Standard **Bag-of-Words** Pipeline (for image classification)

Dictionary Learning:
Learn Visual Words using clustering

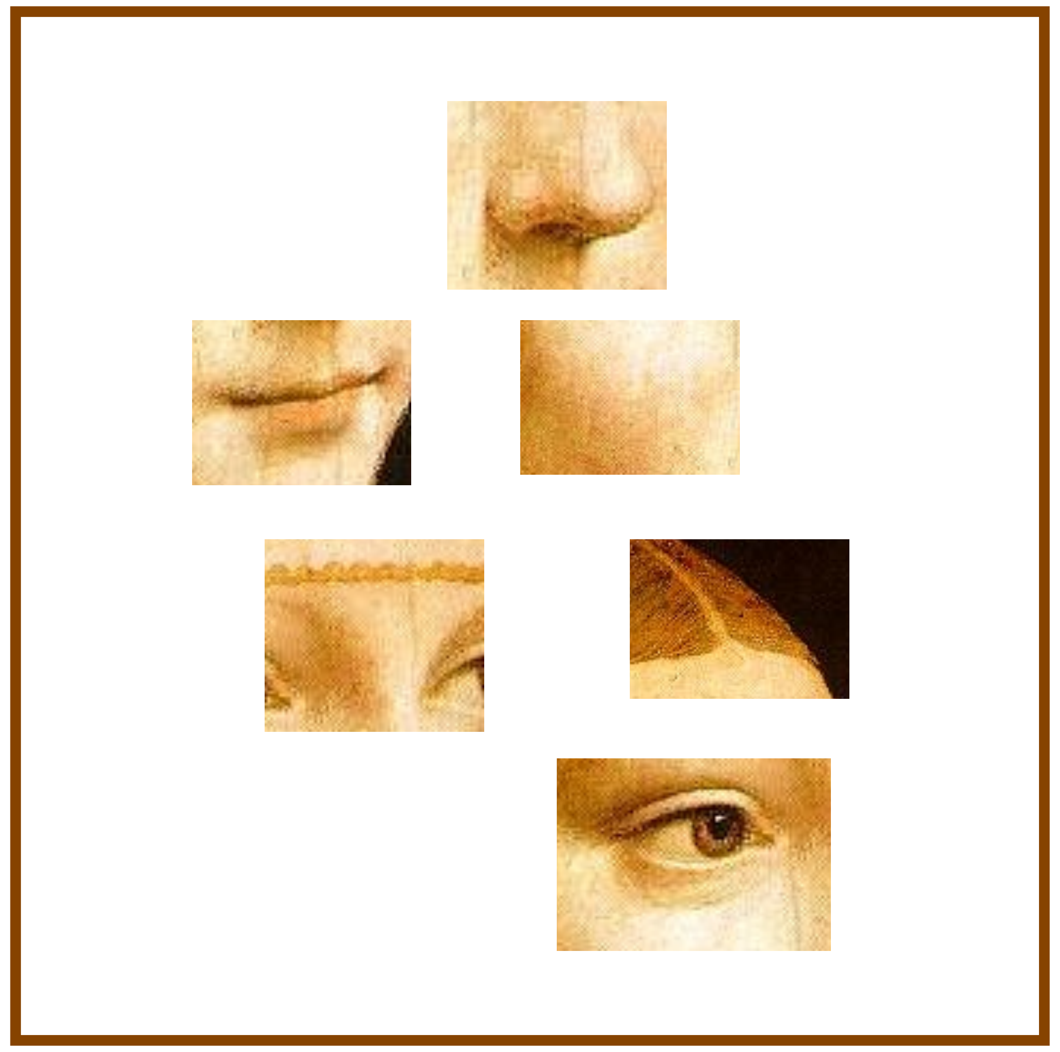
Encode:
build Bags-of-Words (BOW) vectors
for each image

Classify:
Train and test data using BOWs

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

1. **Dictionary Learning**: Learn Visual Words using Clustering

1. **extract features** (e.g., SIFT) from images



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

1. **Dictionary Learning**: Learn Visual Words using Clustering

2. **Learn visual dictionary** (e.g., K-means clustering)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

What **Features** Should We Extract?

— Regular grid

Vogel & Schiele, 2003

Fei-Fei & Perona, 2005

— Interest point detector

Csurka et al. 2004

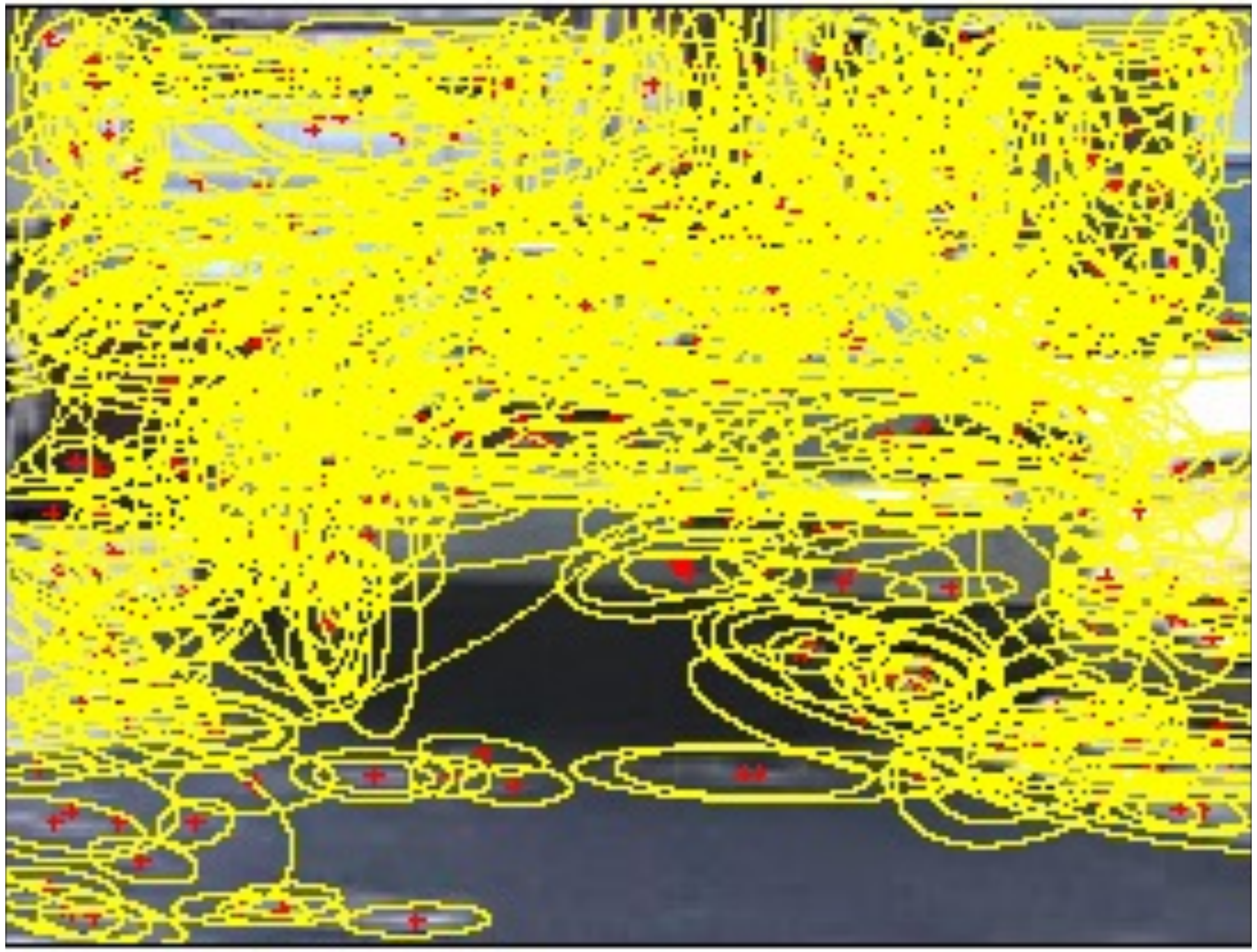
Fei-Fei & Perona, 2005

Sivic et al. 2005

— Other methods

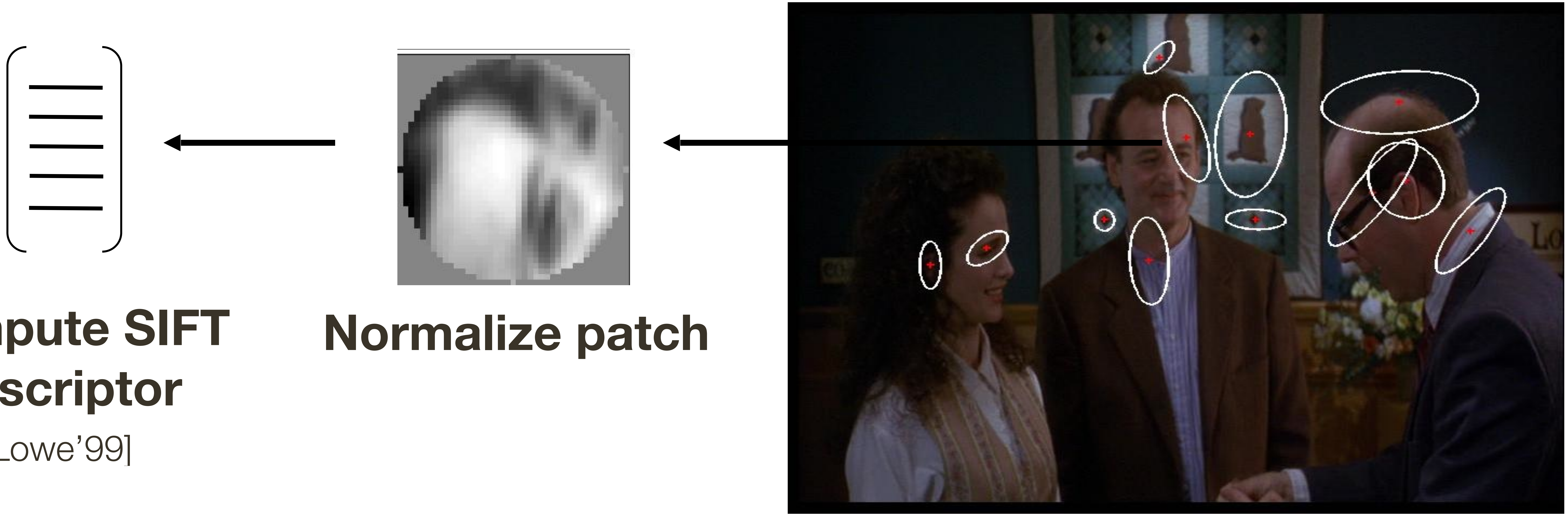
Random sampling (Vidal-Naquet & Ullman, 2002)

Segmentation-based patches (Barnard et al. 2003)



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Extracting **SIFT** Patches



Compute SIFT descriptor

[Lowe'99]

Normalize patch

Detect patches

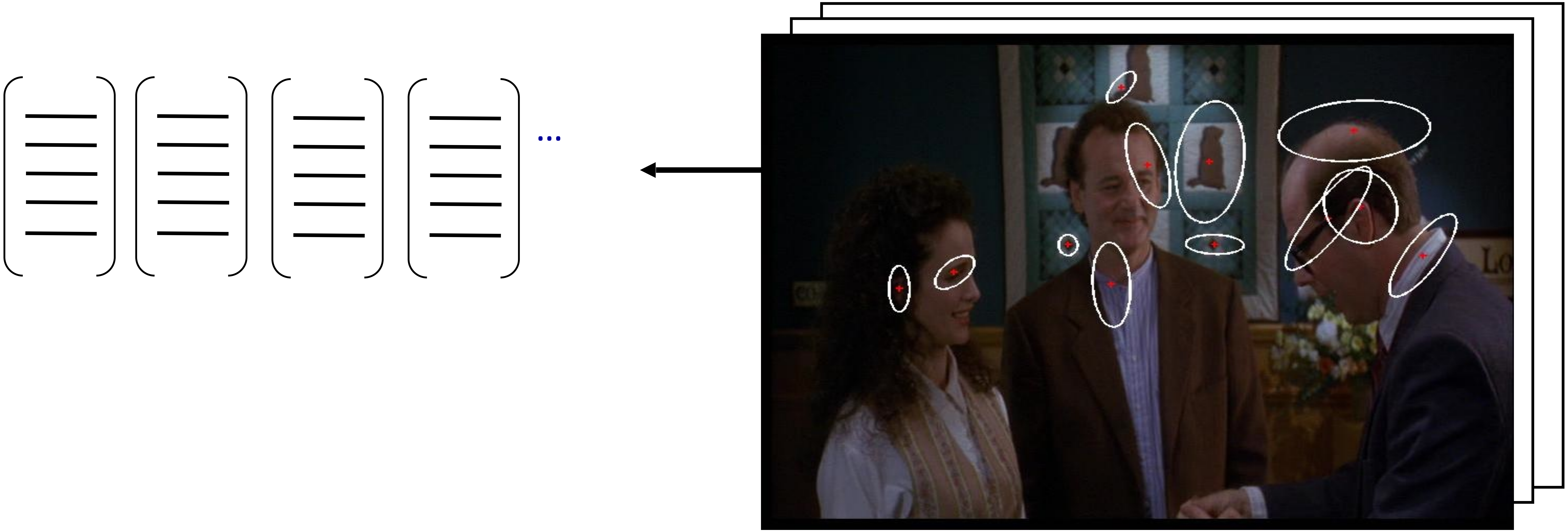
[Mikojaczyk and Schmid '02]

[Mata, Chum, Urban & Pajdla, '02]

[Sivic & Zisserman, '03]

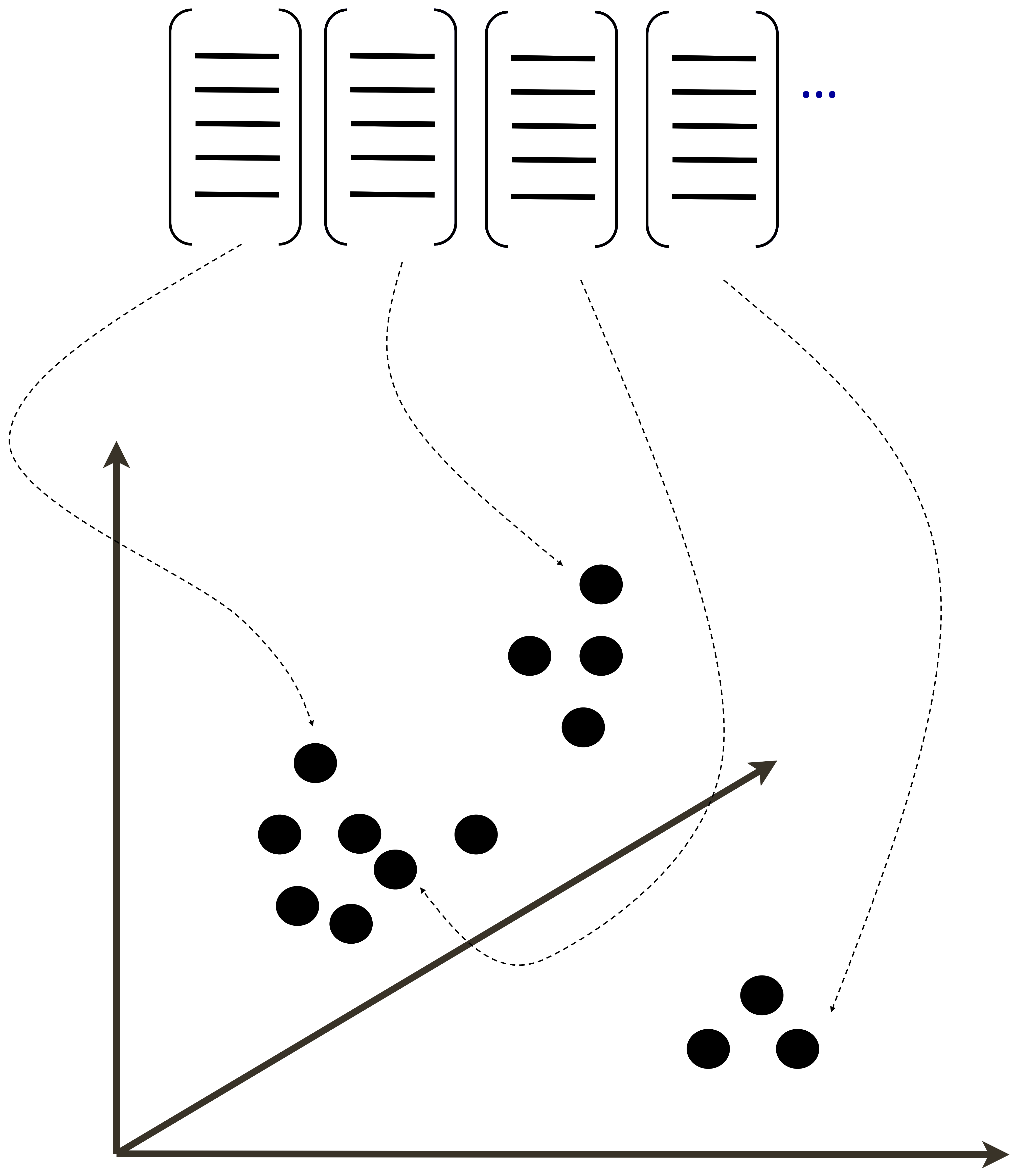
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Extracting **SIFT** Patches



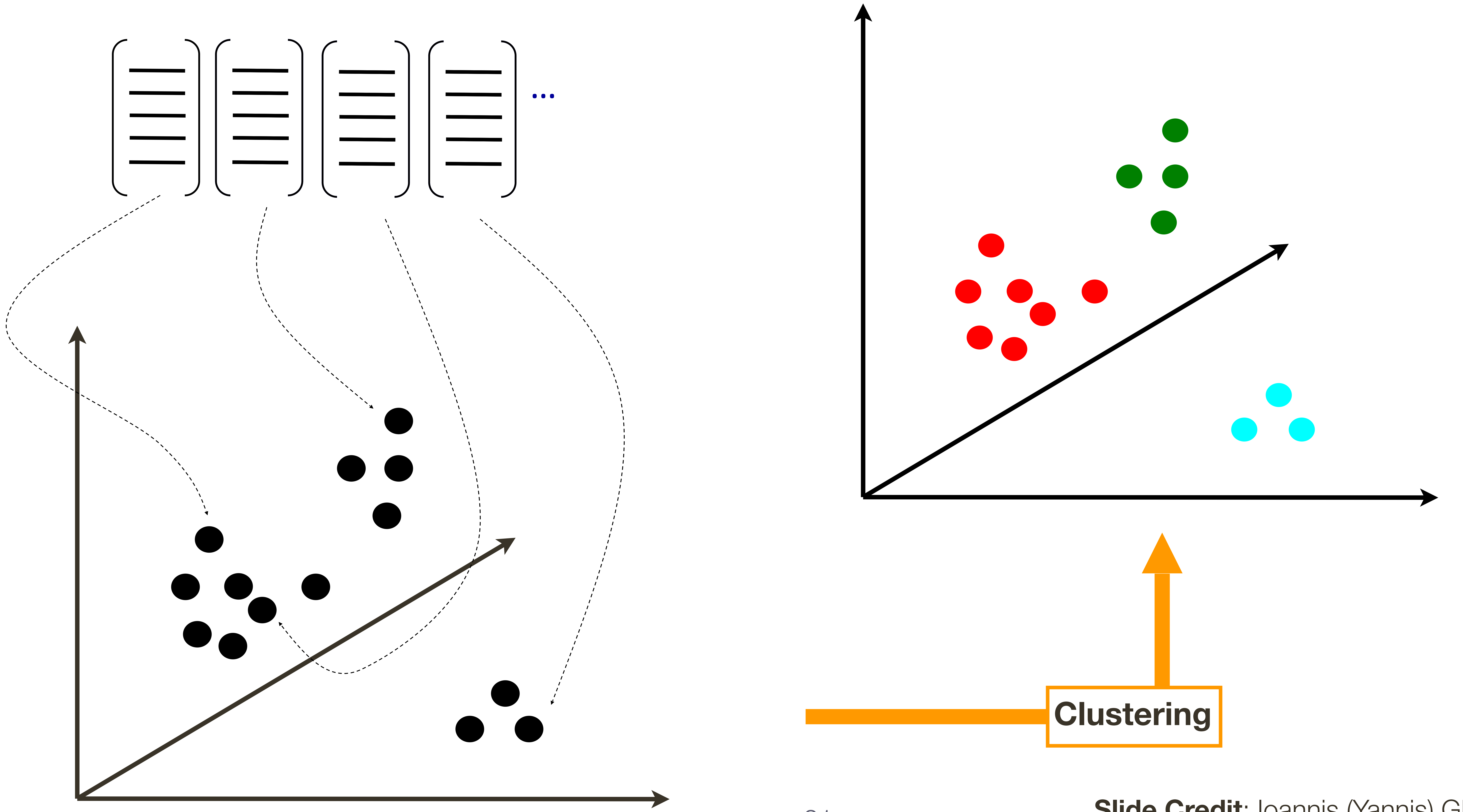
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Creating **Dictionary**



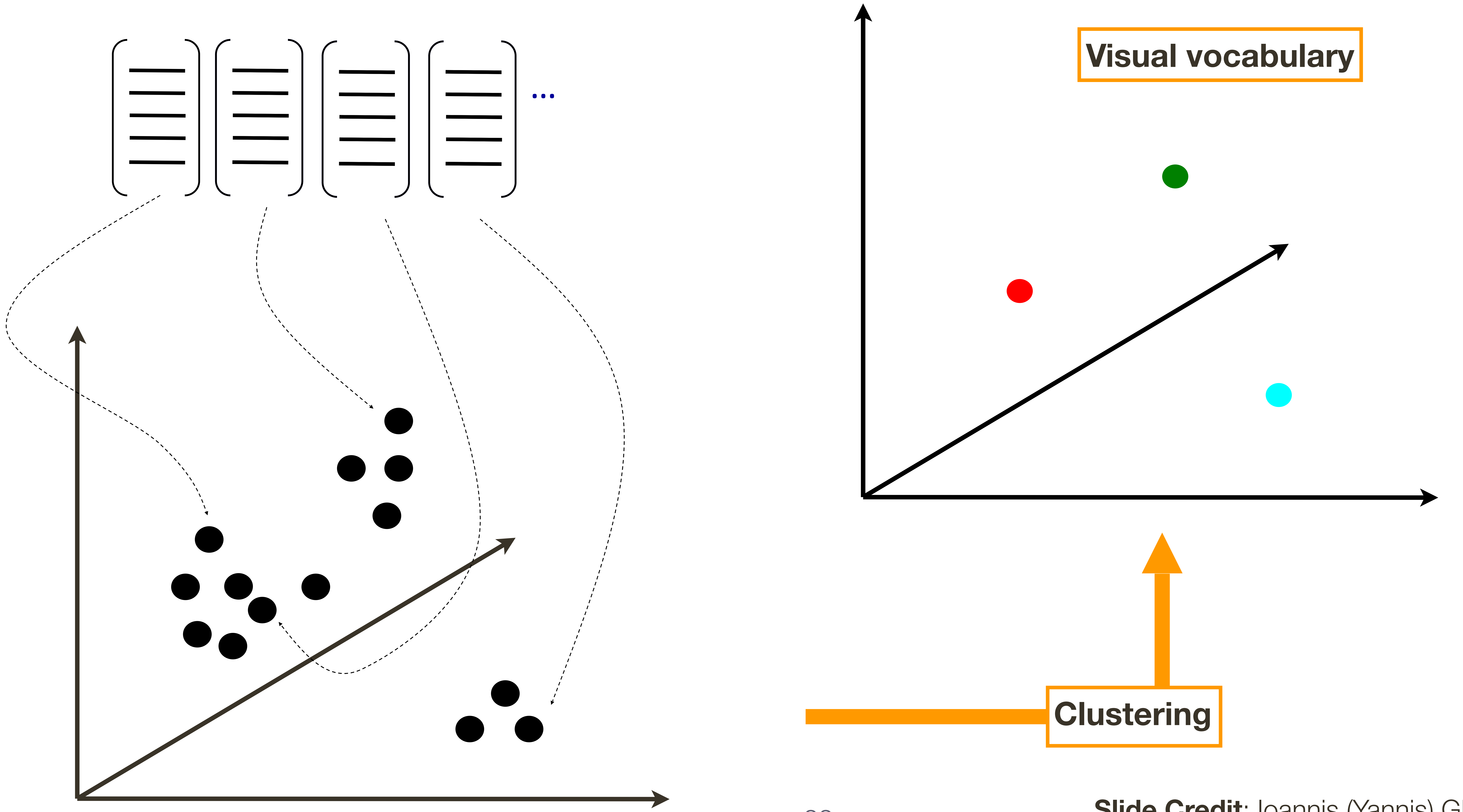
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Creating **Dictionary**



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Creating **Dictionary**



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

K-means clustering

K-means Clustering

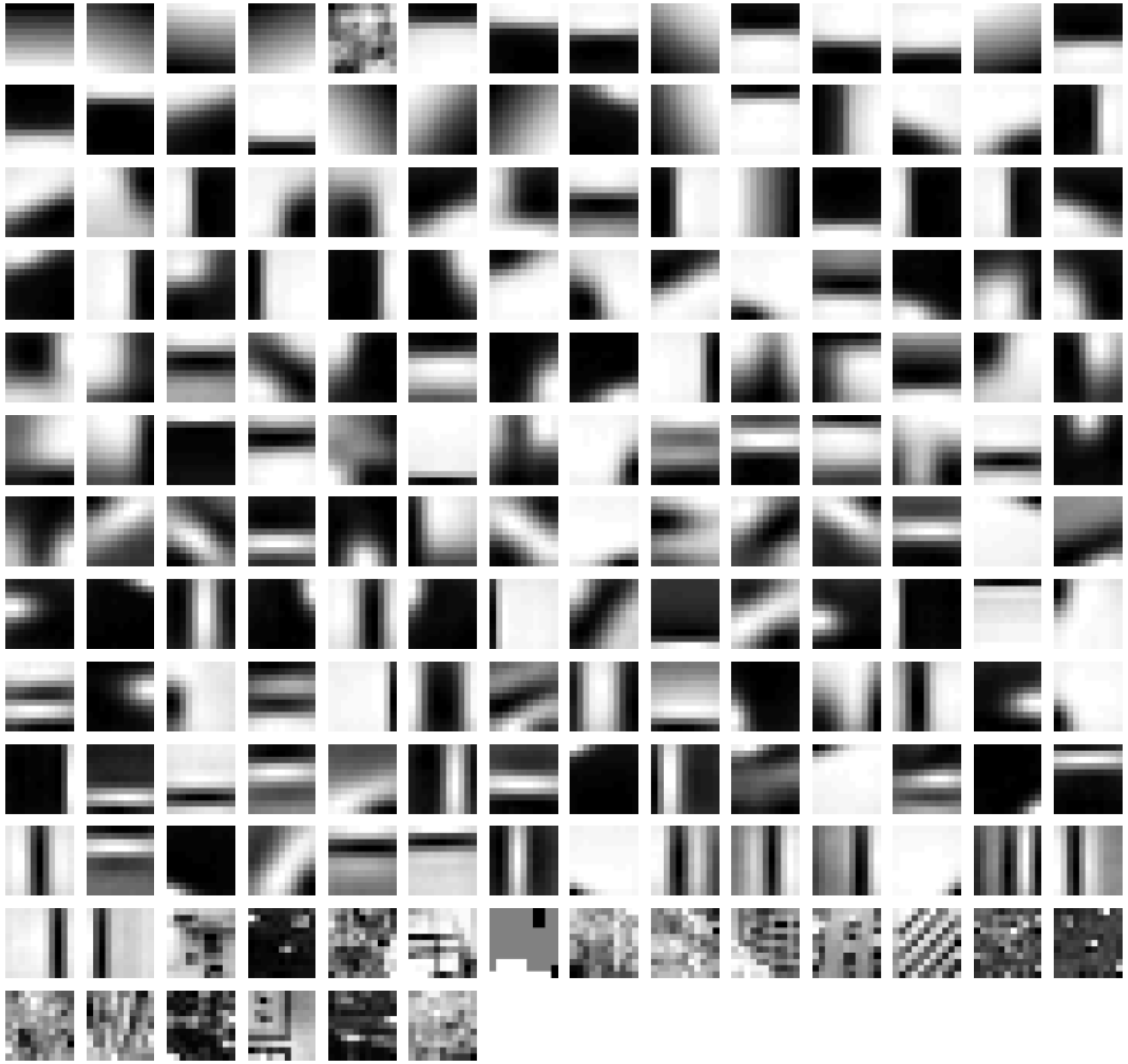
K-means is a clustering technique that iterates between

1. Assume the cluster centers are known. Assign each point to the closest cluster center.

2. Assume the assignment of points to clusters is known. Compute the best cluster center for each cluster (as the mean).

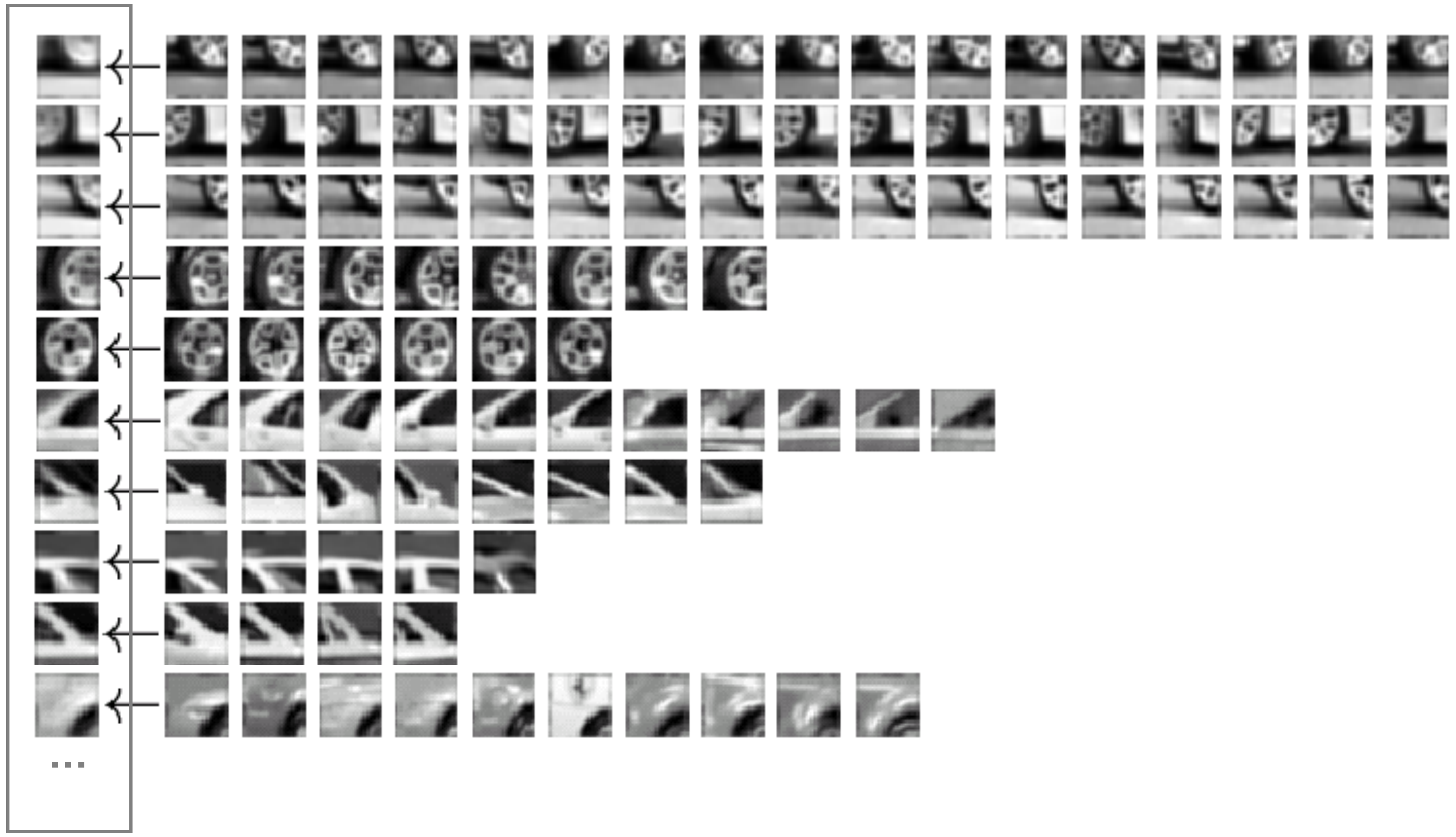
K-means clustering is initialization dependent and converges to a local minimum

Example **Visual Dictionary**



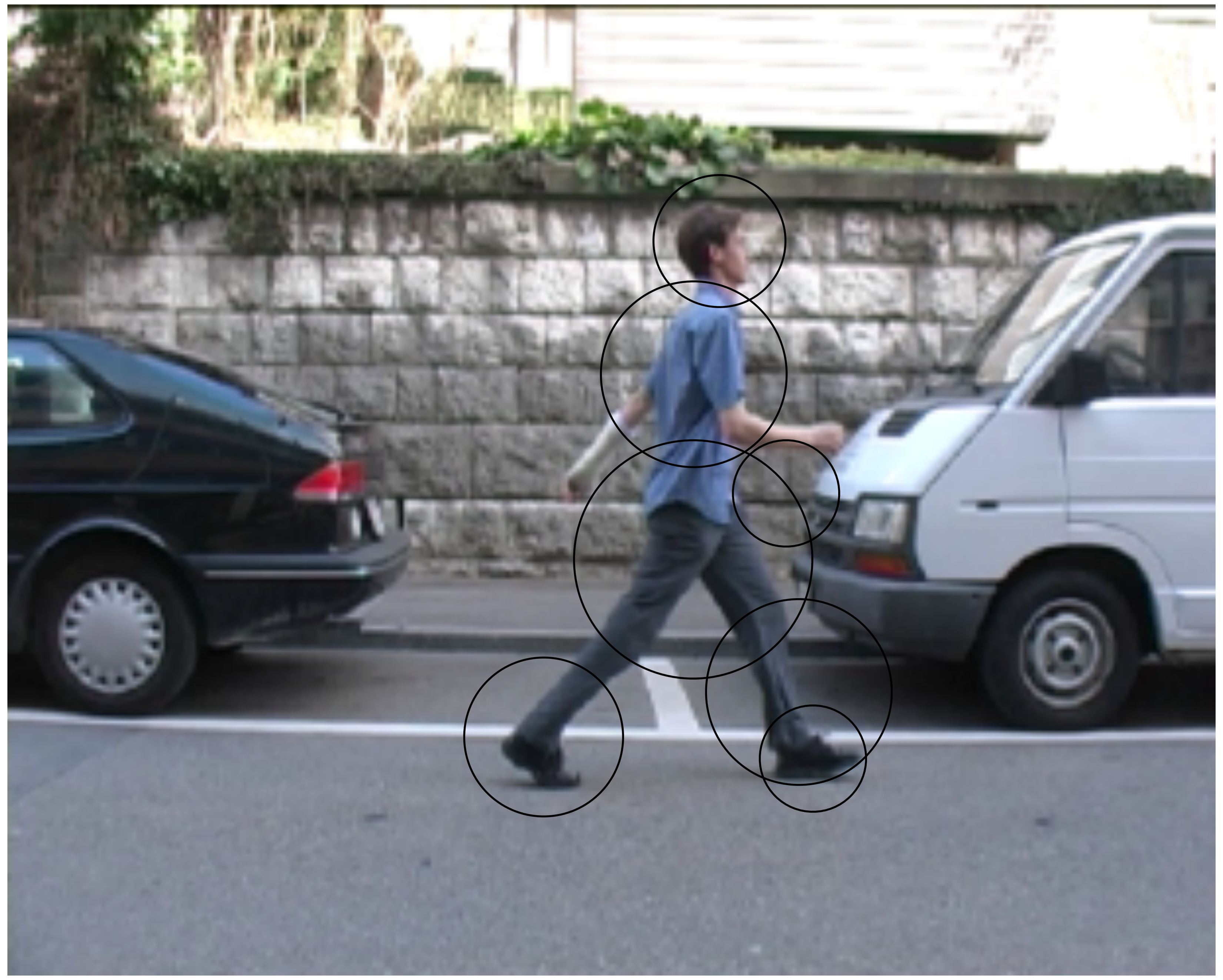
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

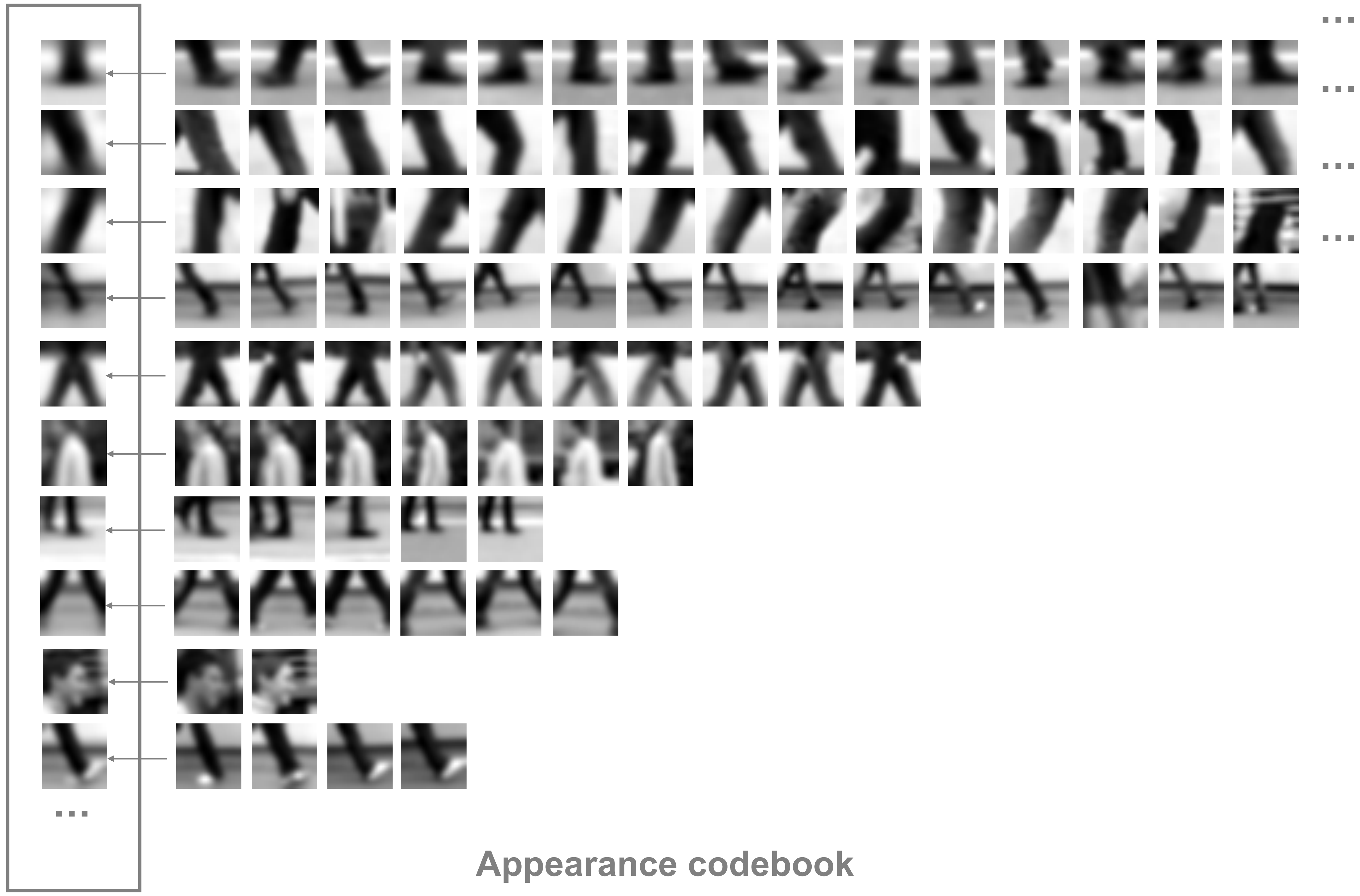
Example **Visual Dictionary**



Source: B. Leibe

Example **Visual Dictionary**





Source: B. Leibe

Standard **Bag-of-Words** Pipeline (for image classification)

Dictionary Learning:
Learn Visual Words using clustering

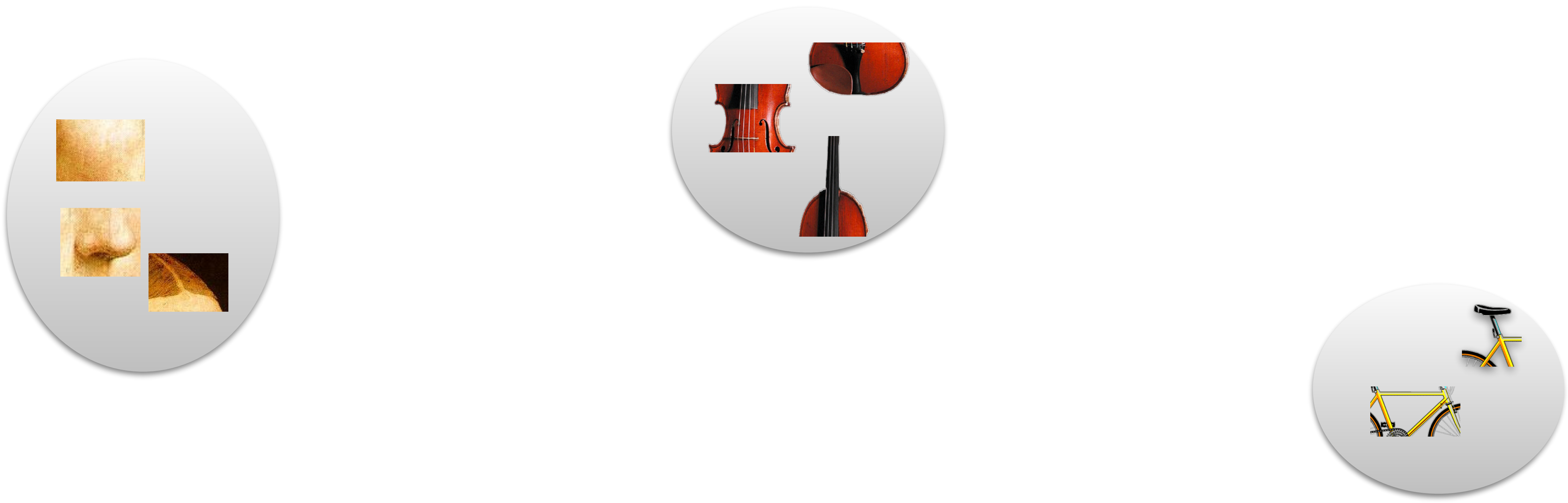
Encode:
build Bags-of-Words (BOW) vectors
for each image

Classify:
Train and test data using BOWs

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

2. **Encode:** build Bag-of-Words (BOW) vectors for each image

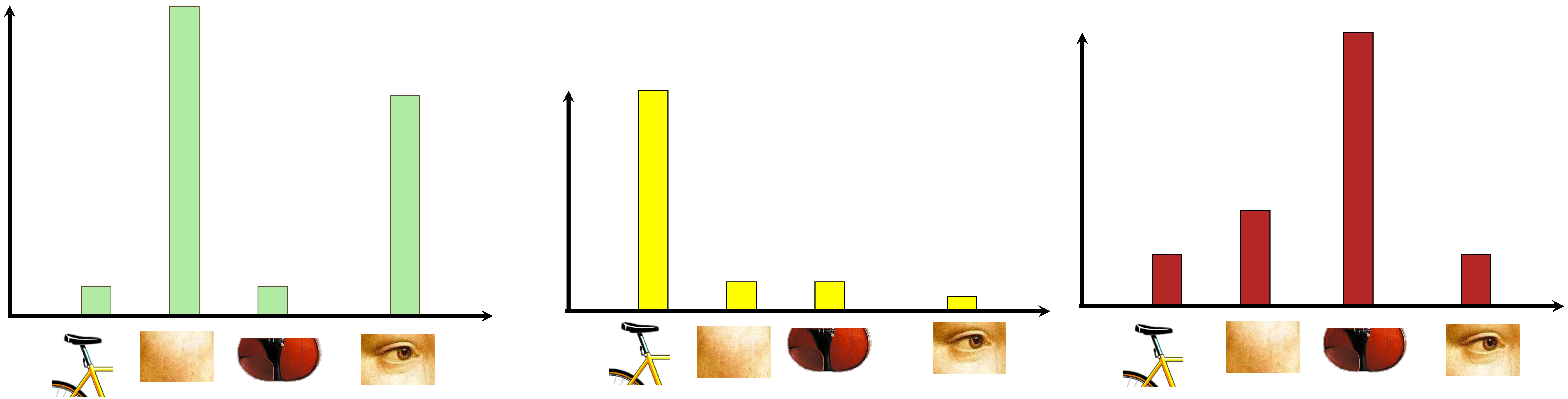
1. **Quantization**: image features gets associated to a visual word (nearest cluster center)



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

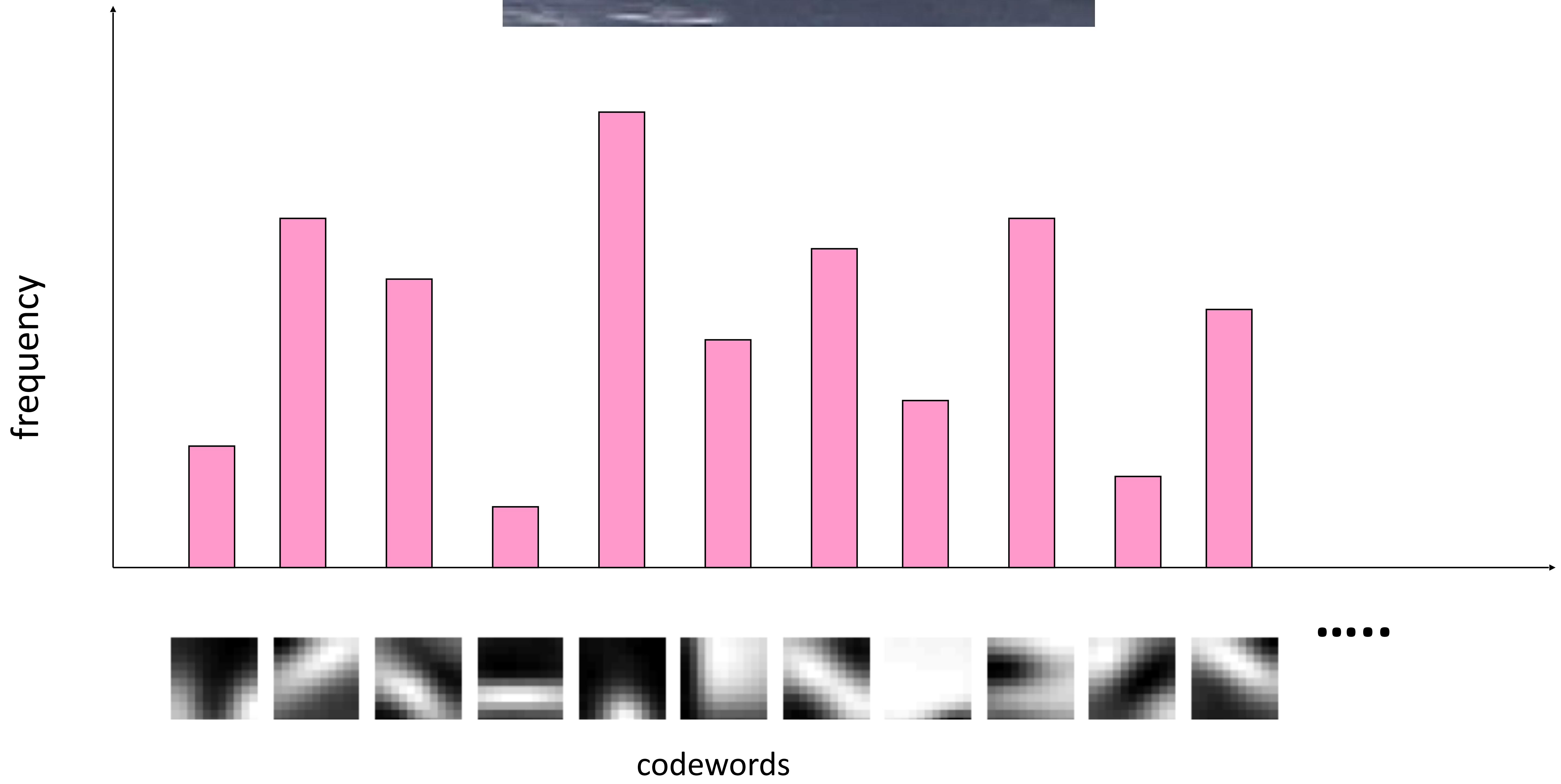
2. Encode: build Bag-of-Words (BOW) vectors for each image

2. **Histogram**: count the number of visual word occurrences



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

2. Encode: build Bag-of-Words (BOW) vectors for each image



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Standard **Bag-of-Words** Pipeline (for image classification)

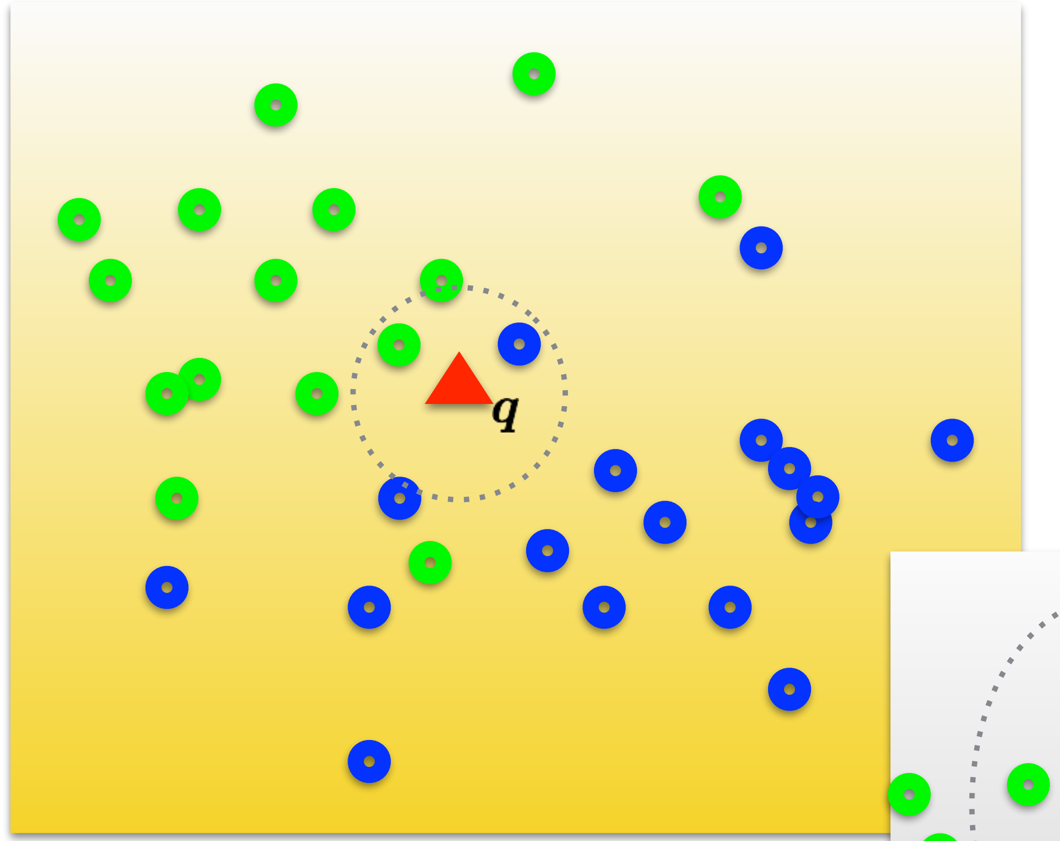
Dictionary Learning:
Learn Visual Words using clustering

Encode:
build Bags-of-Words (BOW) vectors
for each image

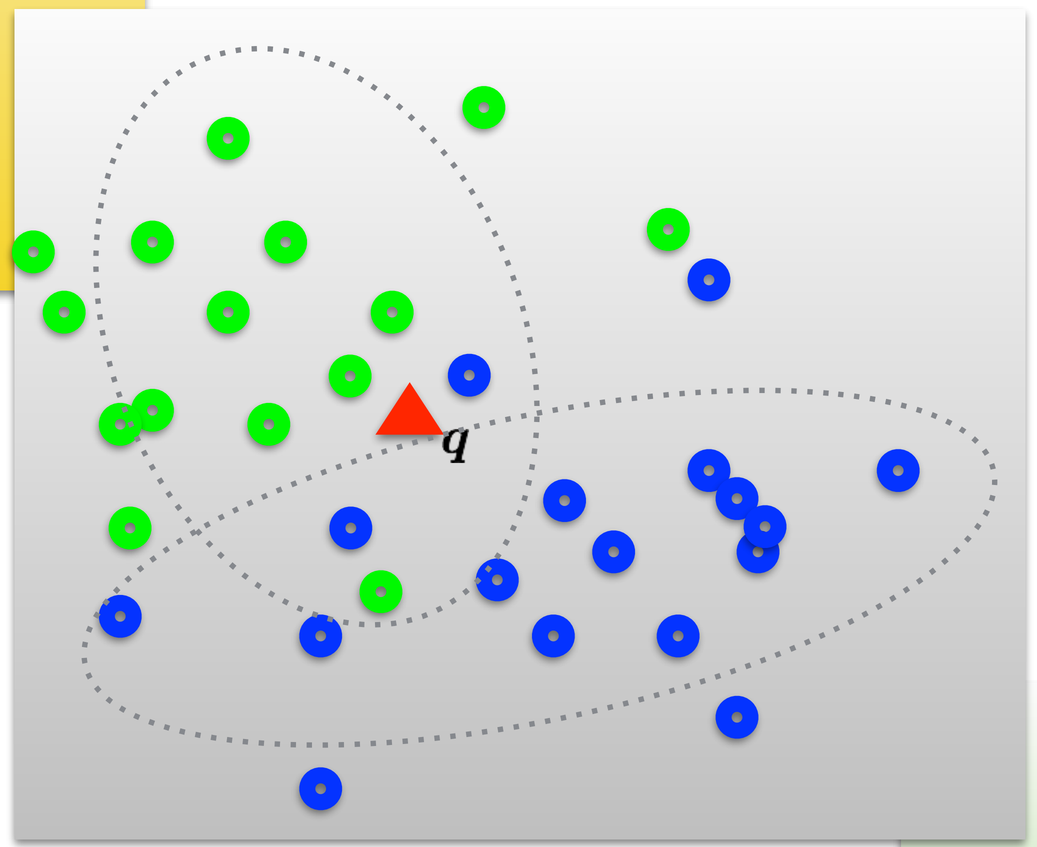
Classify:
Train and test data using BOWs

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

3. **Classify**: Train and text classifier using BOWs

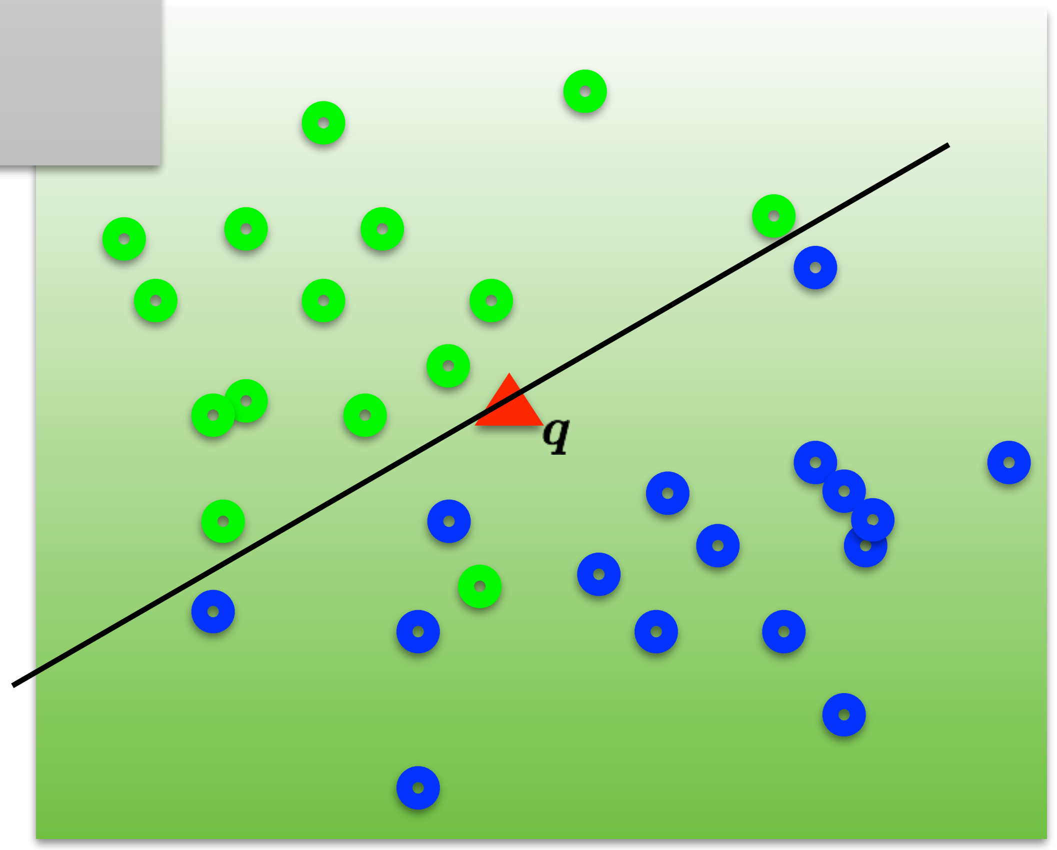


K nearest neighbors



Naïve Bayes

Support Vector Machine



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Bag-of-Words Representation

Algorithm:

Initialize an empty K -bin histogram, where K is the number of codewords
Extract local descriptors (e.g. SIFT) from the image
For each local descriptor **x**

Map (Quantize) **x** to its closest codeword → **c**(**x**)

Increment the histogram bin for **c**(**x**)

Return histogram

We can then classify the histogram using a trained classifier, e.g. a support vector machine or k-Nearest Neighbor classifier

Spatial Pyramid

The bag of words representation does not preserve any spatial information

The **spatial pyramid** is one way to incorporate spatial information into the image descriptor.

A spatial pyramid partitions the image and counts codewords within each grid box; this is performed at multiple levels

Spatial Pyramid

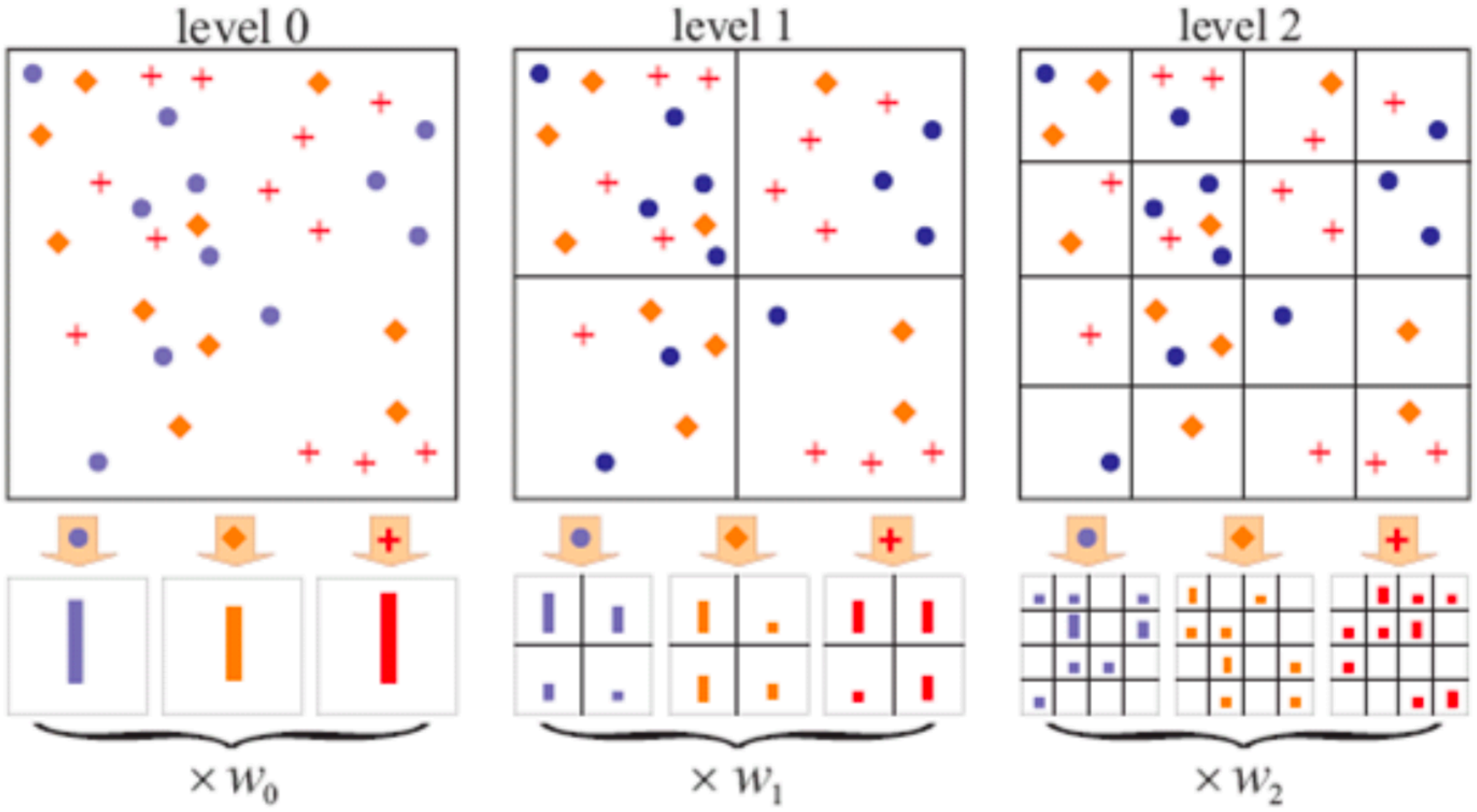


Fig. 16.8 in Forsyth & Ponce (2nd ed.).
Original credit: Lazebnik et al., 2006

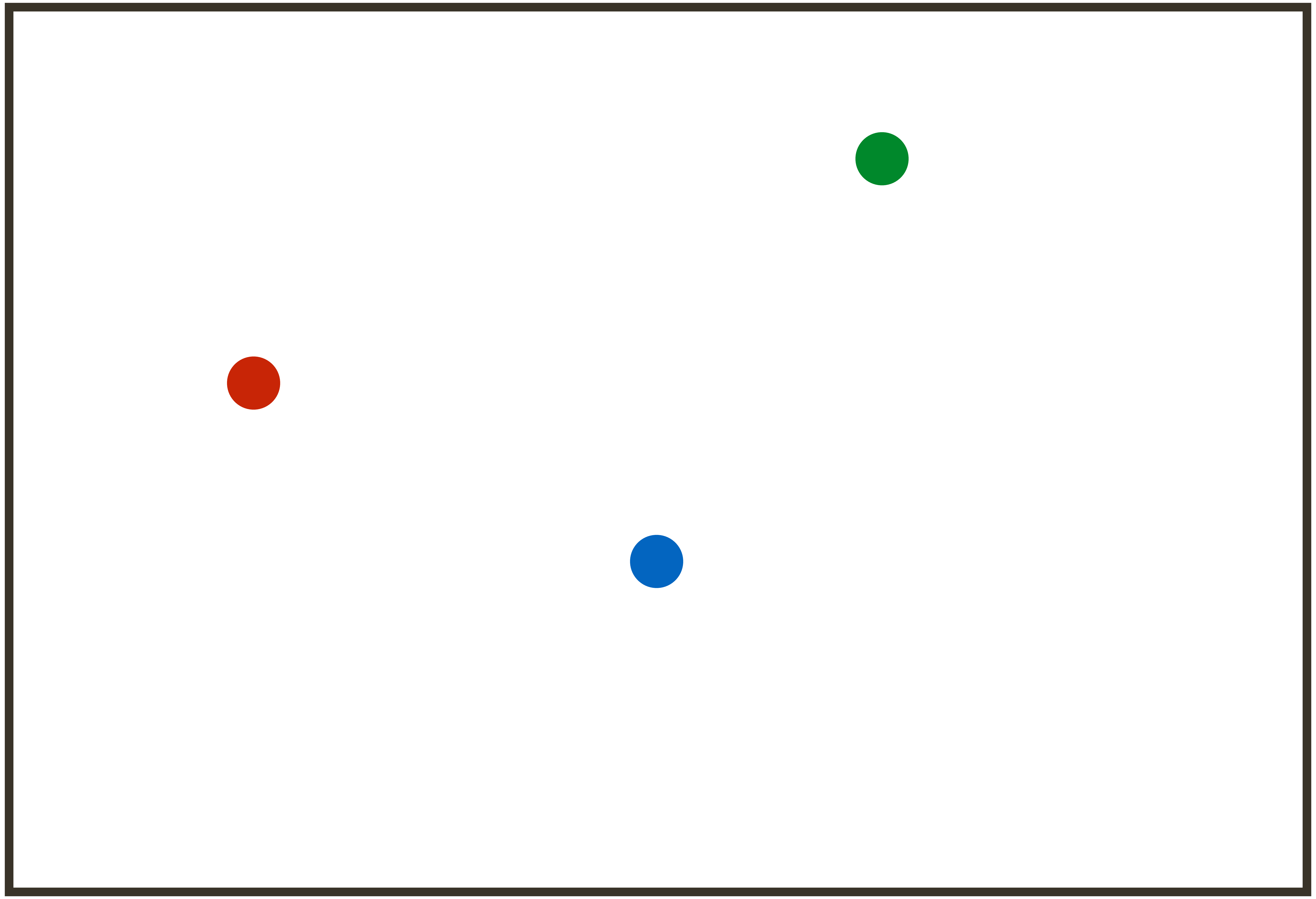
VLAD (Vector of Locally Aggregated Descriptors)

There are more advanced ways to ‘count’ visual words than incrementing its histogram bin

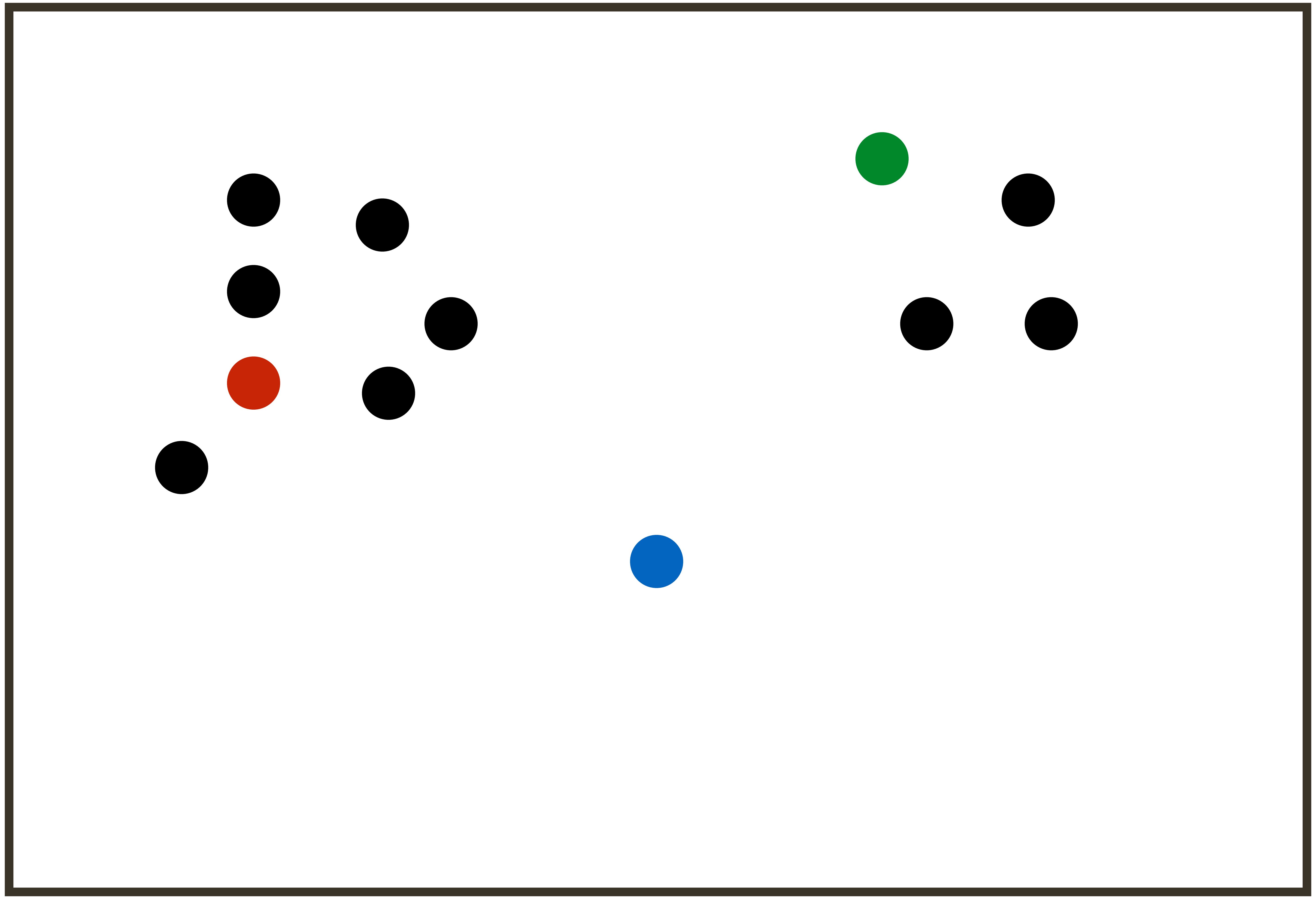
For example, it might be useful to describe how local descriptors are quantized to their visual words

In the VLAD representation, instead of incrementing the histogram bin by one, we increment it by the **residual** vector $\boldsymbol{x} - \boldsymbol{c}(\boldsymbol{x})$

Example: VLAD

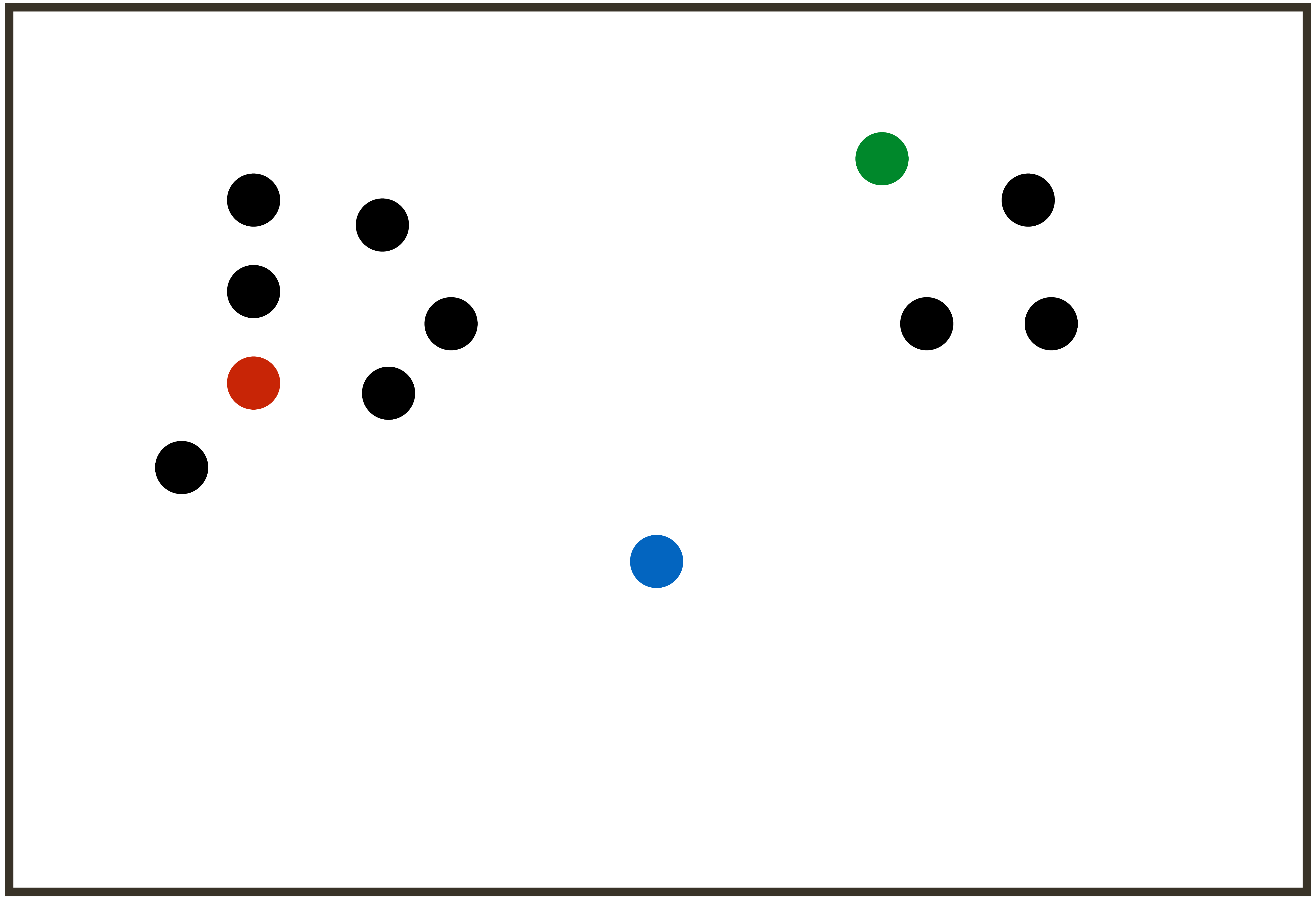


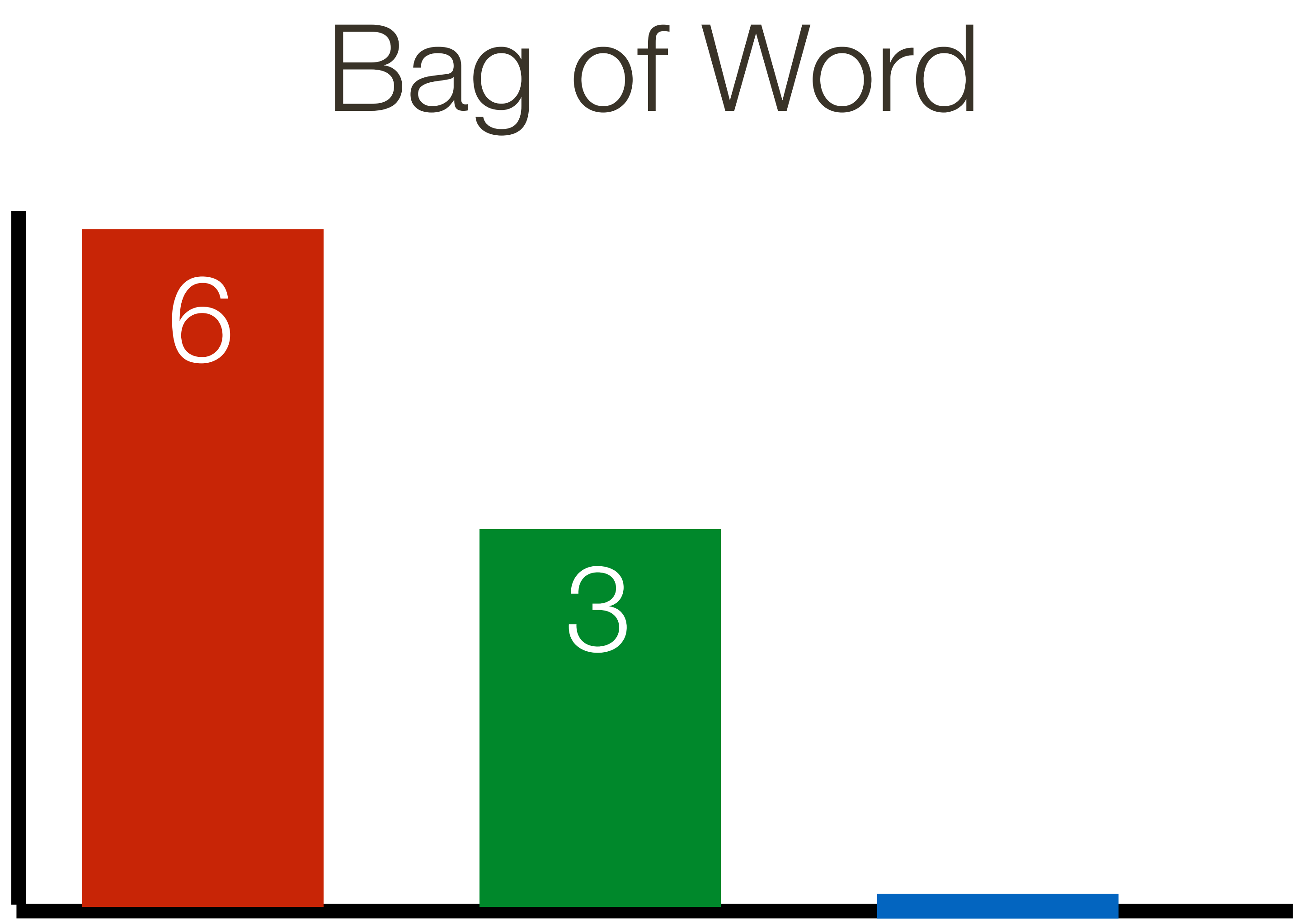
Example: VLAD



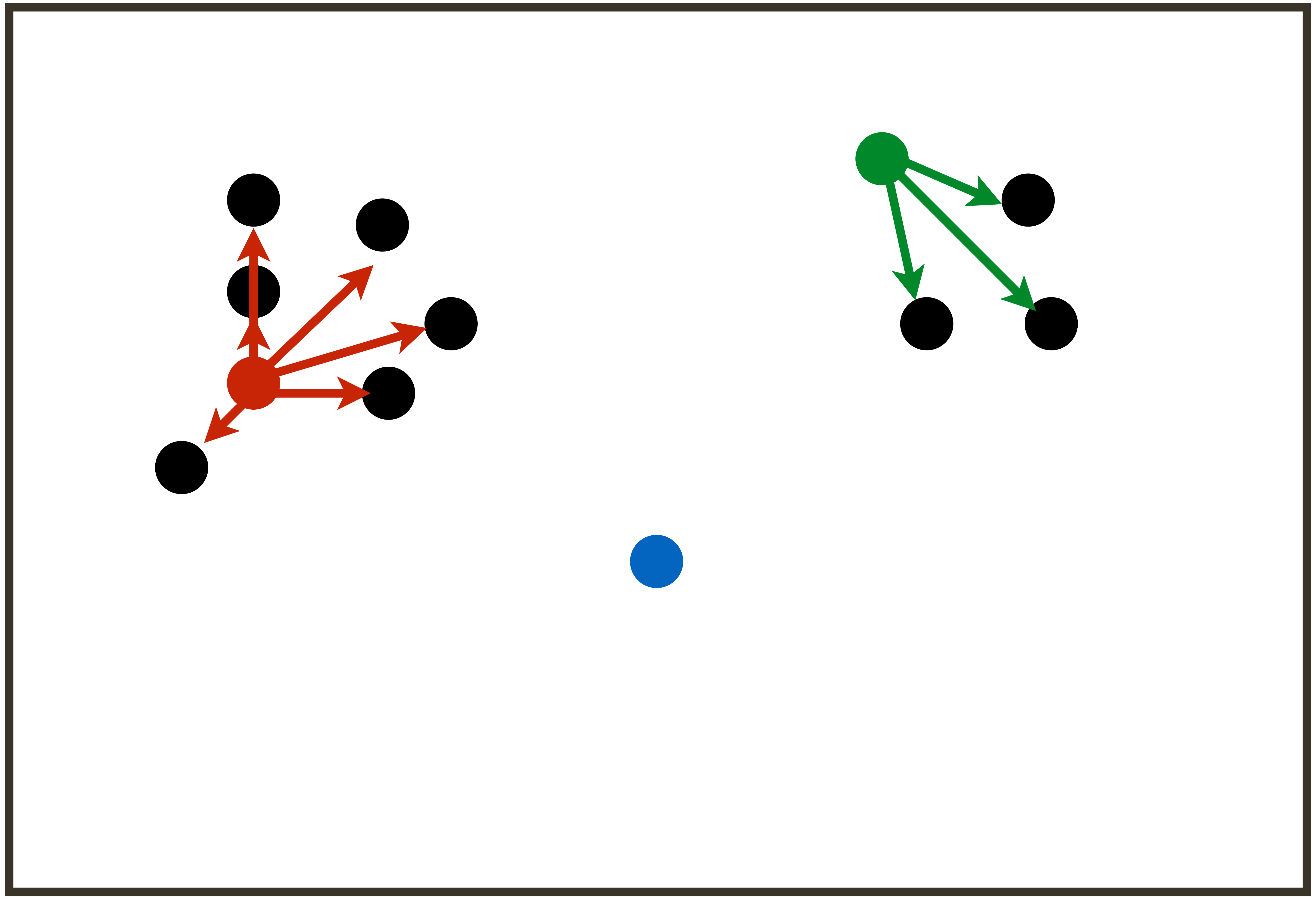
Bag of Word

Example: VLAD

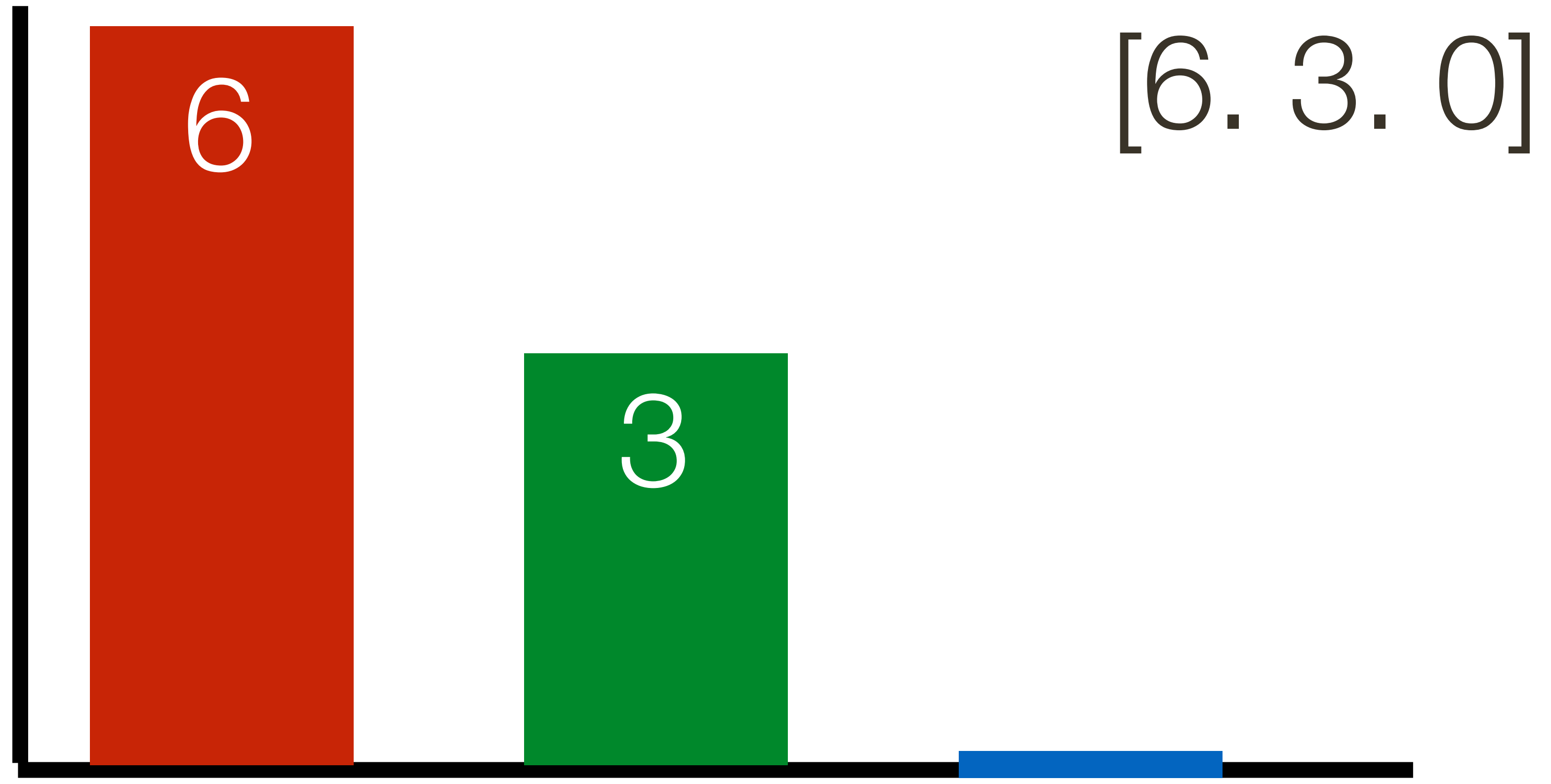




Example: VLAD

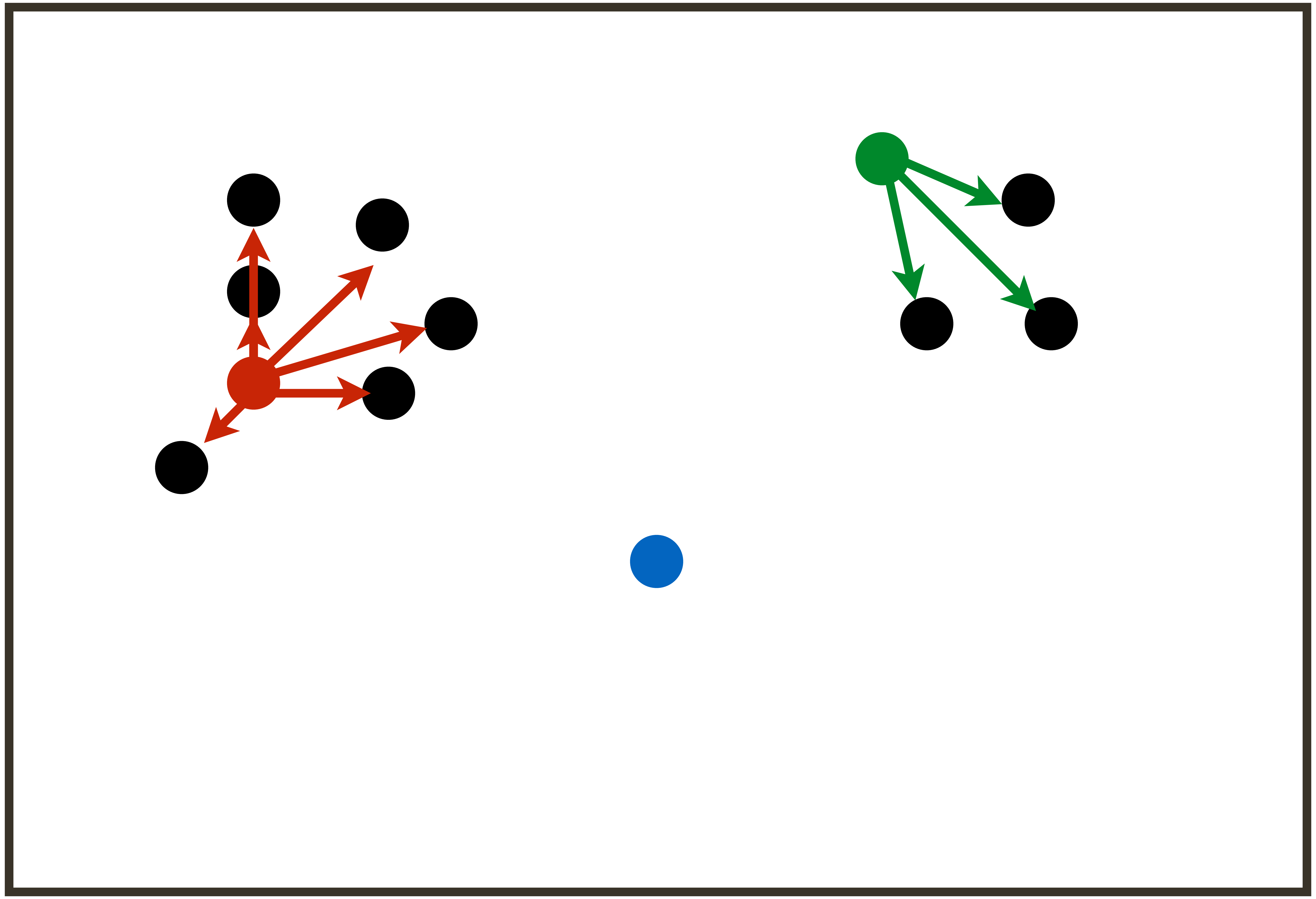


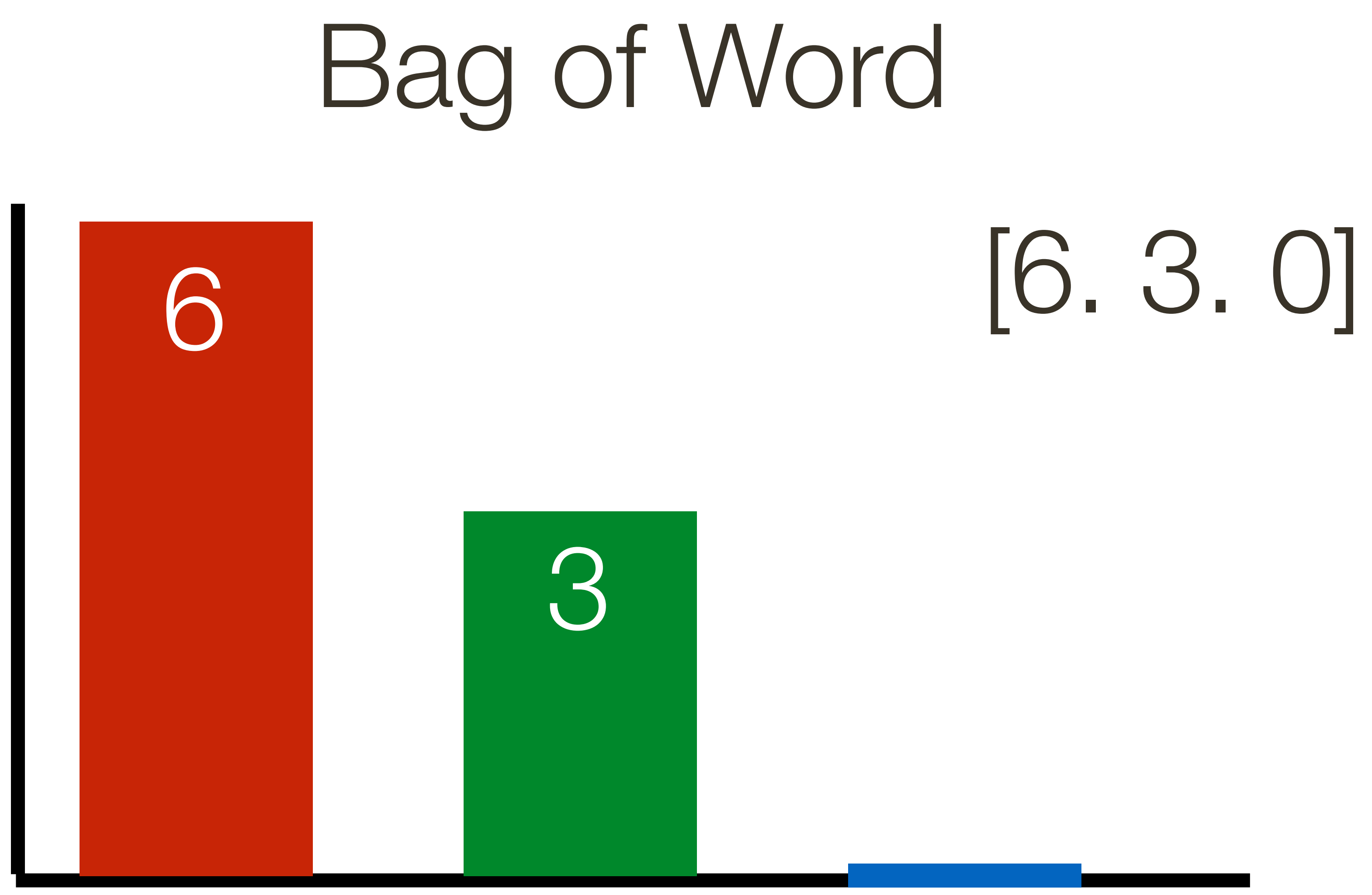
Bag of Word

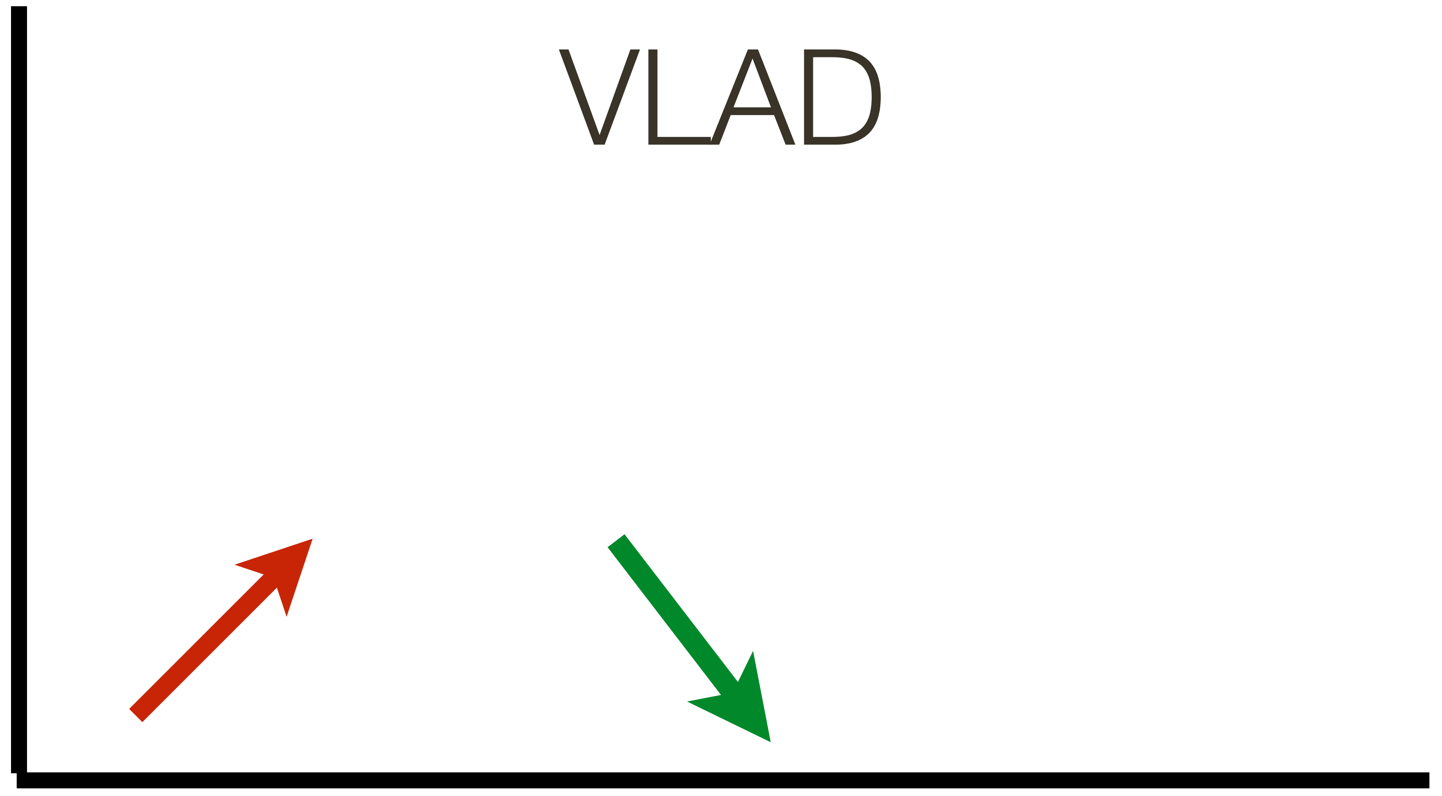


VLAD

Example: VLAD







VLAD (Vector of Locally Aggregated Descriptors)

The dimensionality of a **VLAD** descriptor is Kd

— K : number of codewords

— d : dimensionality of the local descriptor

VLAD characterizes the distribution of local descriptors with respect to the codewords

Summary

Factors that make image classification hard
— intra-class variation, viewpoint, illumination, clutter, occlusion...

A codebook of **visual words** contains representative local patch descriptors
— can be constructed by clustering local descriptors (e.g. SIFT) in training images

The **bag of words** model accumulates a histogram of occurrences of each visual word

The **spatial pyramid** partitions the image and counts visual words within each grid box; this is repeated at multiple levels

Back to **Classification**

Decision Tree

A **decision tree** is a simple non-linear parametric classifier

Consists of a tree in which each internal node is associated with a feature test

A data point starts at the root and recursively proceeds to the child node determined by the feature test, until it reaches a leaf node

The leaf node stores a class label or a probability distribution over class labels

Decision Tree

Learning a decision tree from a training set involves selecting an efficient sequence of feature tests

Example: Waiting for a restaurant table

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*T*	*F*	*F*	*T*	*Some*	*$$$*	*F*	*T*	*French*	*0–10*	*T*
X_2	*T*	*F*	*F*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*30–60*	*F*
X_3	*F*	*T*	*F*	*F*	*Some*	*$*	*F*	*F*	*Burger*	*0–10*	*T*
X_4	*T*	*F*	*T*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*10–30*	*T*
X_5	*T*	*F*	*T*	*F*	*Full*	*$$$*	*F*	*T*	*French*	*>60*	*F*
X_6	*F*	*T*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Italian*	*0–10*	*T*
X_7	*F*	*T*	*F*	*F*	*None*	*$*	*T*	*F*	*Burger*	*0–10*	*F*
X_8	*F*	*F*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Thai*	*0–10*	*T*
X_9	*F*	*T*	*T*	*F*	*Full*	*$*	*T*	*F*	*Burger*	*>60*	*F*
X_{10}	*T*	*T*	*T*	*T*	*Full*	*$$$*	*F*	*T*	*Italian*	*10–30*	*F*
X_{11}	*F*	*F*	*F*	*F*	*None*	*$*	*F*	*F*	*Thai*	*0–10*	*F*
X_{12}	*T*	*T*	*T*	*T*	*Full*	*$*	*F*	*F*	*Burger*	*30–60*	*T*

Decision Tree

Which test is more helpful?

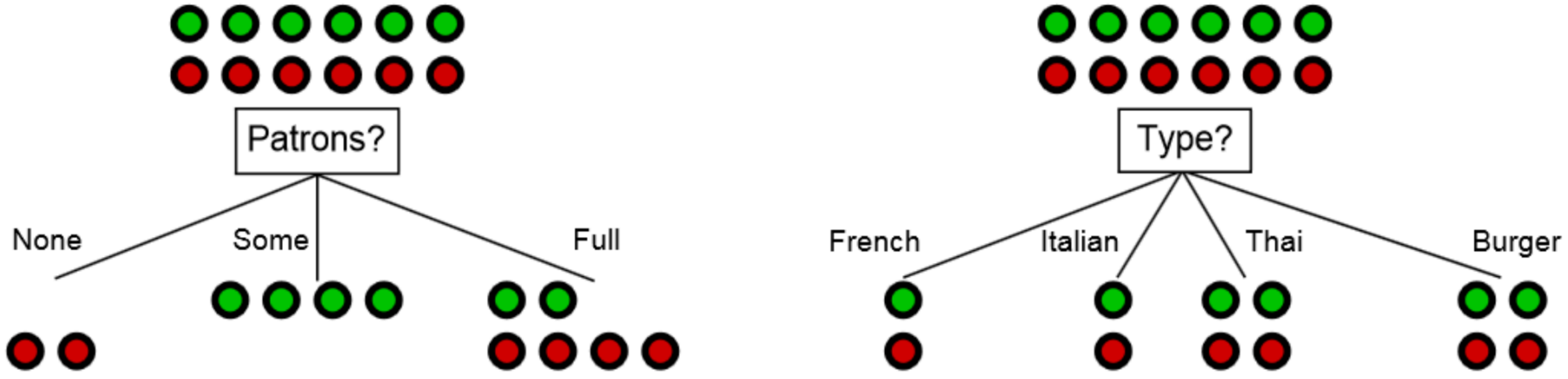


Figure credit: Russell and Norvig (3rd ed.)

Decision Tree

The **entropy** of a set S of data samples is defined as

$$H(S) = -\sum_{c \in C} p(c) \log(p(c))$$

where C is the set of classes represented in S, and $p(c)$ is the empirical distribution of class c in S

Entropy is highest when data samples are spread equally across all classes, and zero when all data samples are from the same class.

Decision Tree

In general we try to select the feature test that maximizes the i**nformation gain**:

$$I = H(S) - \sum_{i \in \{children\}} \frac{|S^i|}{|S|} H(S^i)$$

In the previous example, the information gains of the two candidate tests are:

$$I_{Patrons} = 0.541 \qquad I_{Type} = 0$$

So we choose the ‘Patrons’ test.

Decision Tree

Following this construction procedure we obtain the final decision tree:

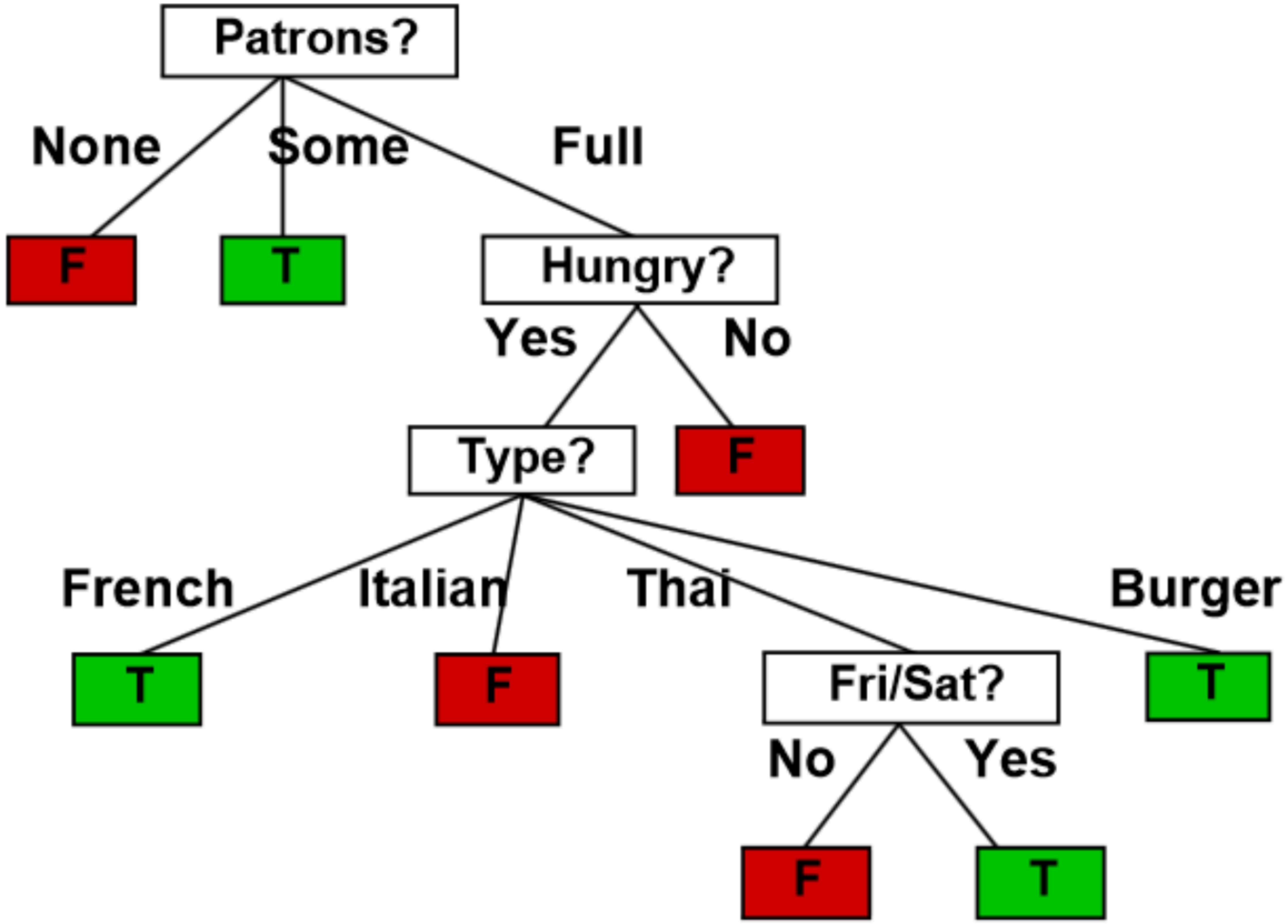


Figure credit: Russell and Norvig (3rd ed.)