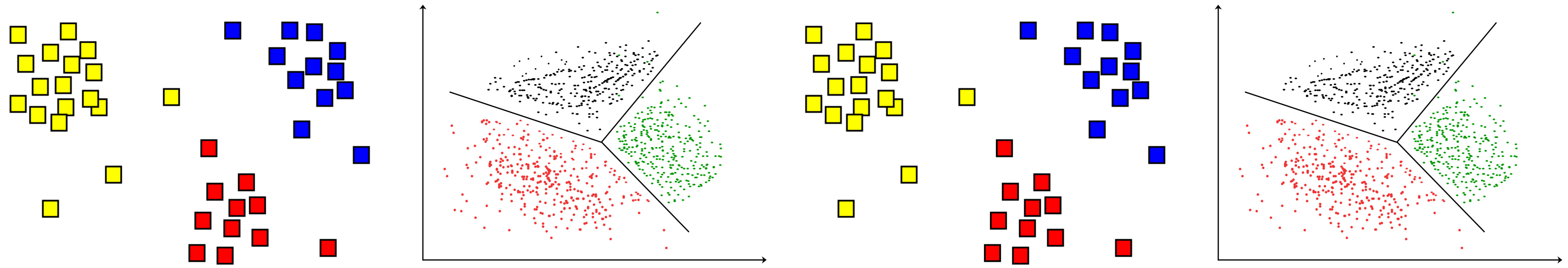
CPSC 425: Computer Vision



Lecture 27: Image Classification (part 3)

Menu for Today (**November 16, 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:

- **Assignment 5**: Scene Recognition with Bag of Words due **November 20**
- **Quiz 5** on Wednsday

Lecture 26: Re-cap

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

Bag-of-Words Representation

Dictionary Learning:
Learn Visual Words using clustering

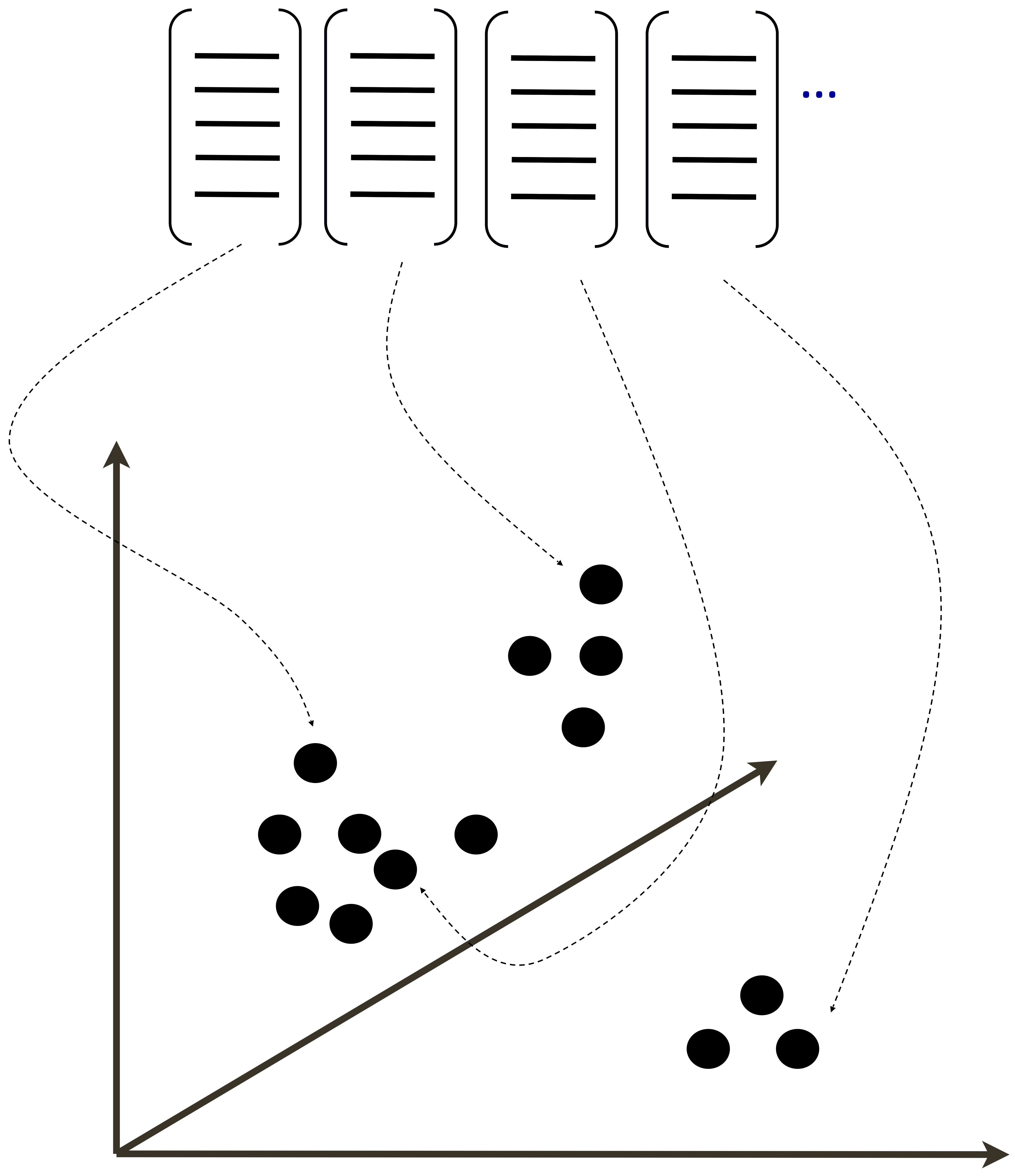
Encode:
build Bags-of-Words (BOW) vectors
for each image

Classify:
Train and test data using BOWs

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 26: Re-cap

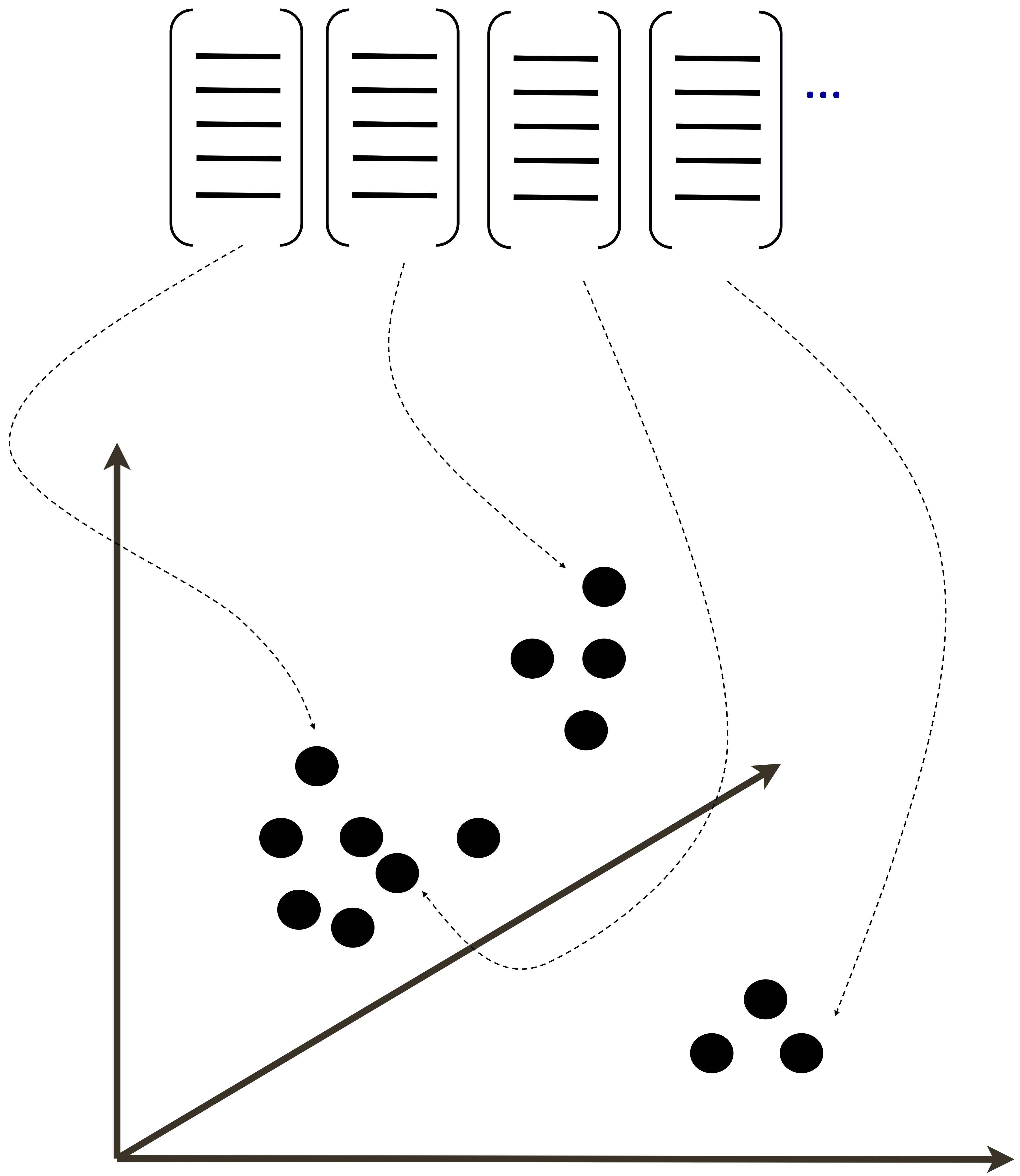
Bag-of-Words Representation

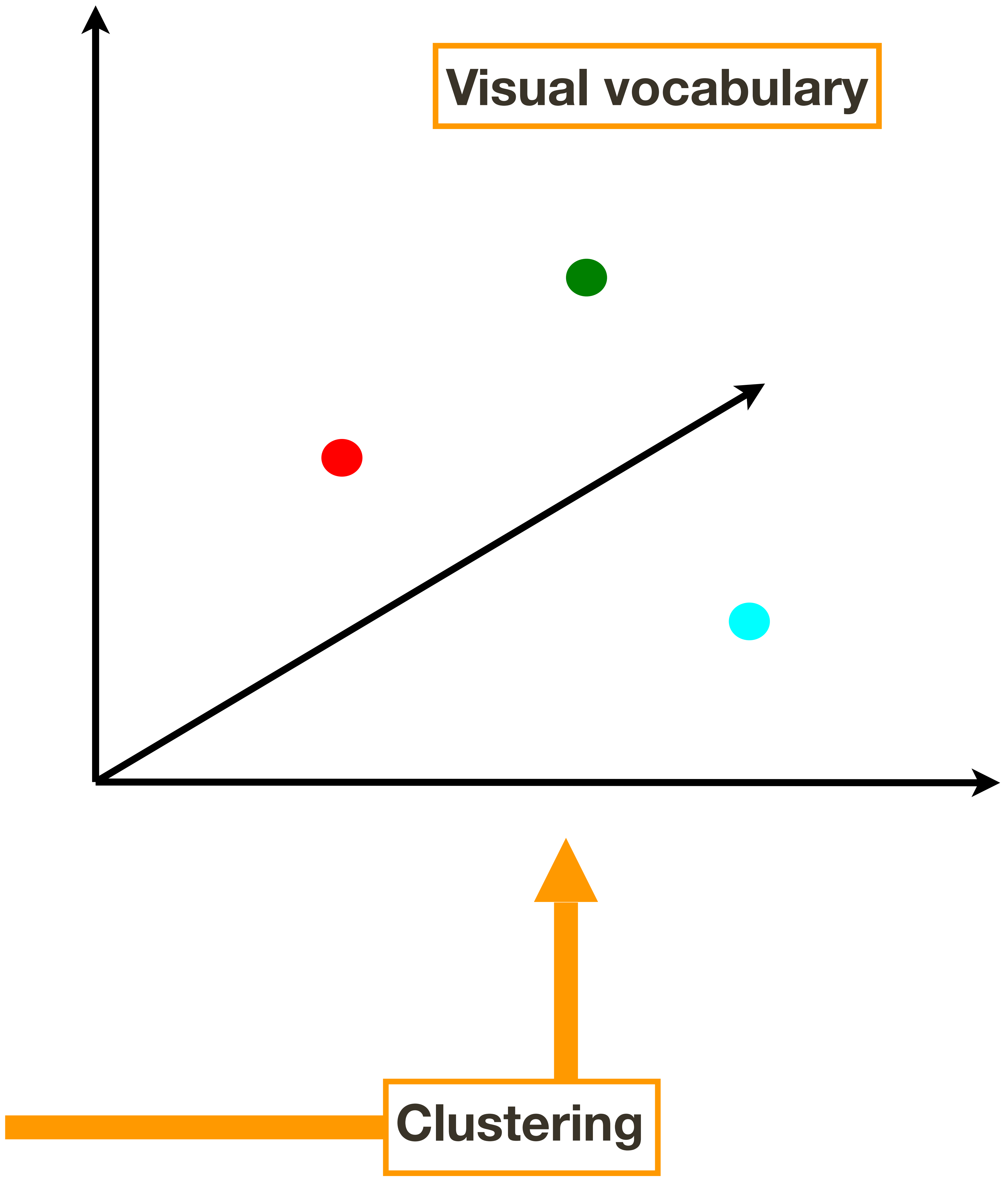


Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 26: Re-cap

Bag-of-Words Representation



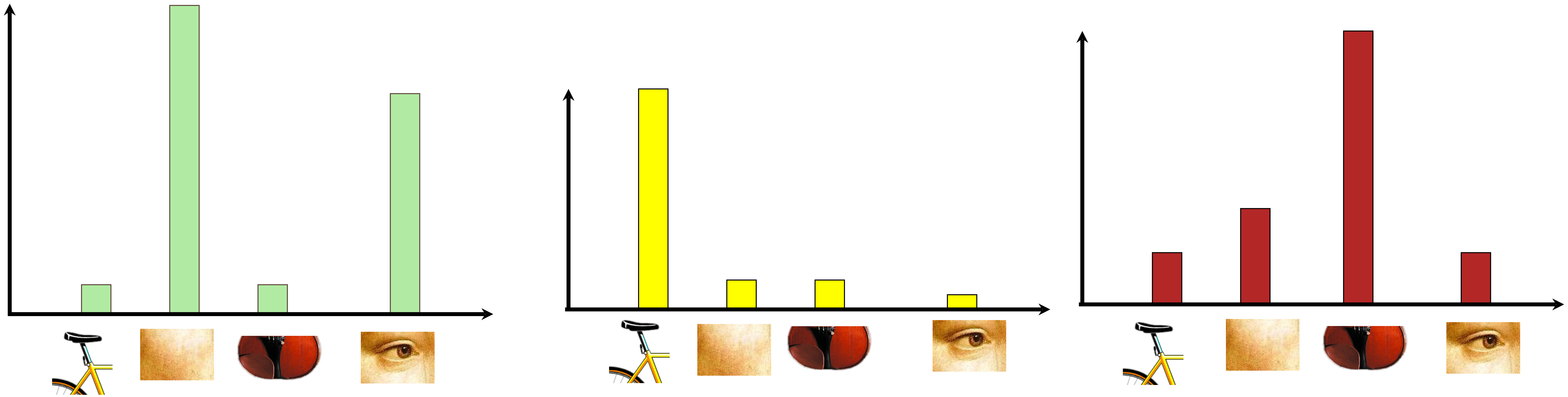


Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Bag-of-Words Representation

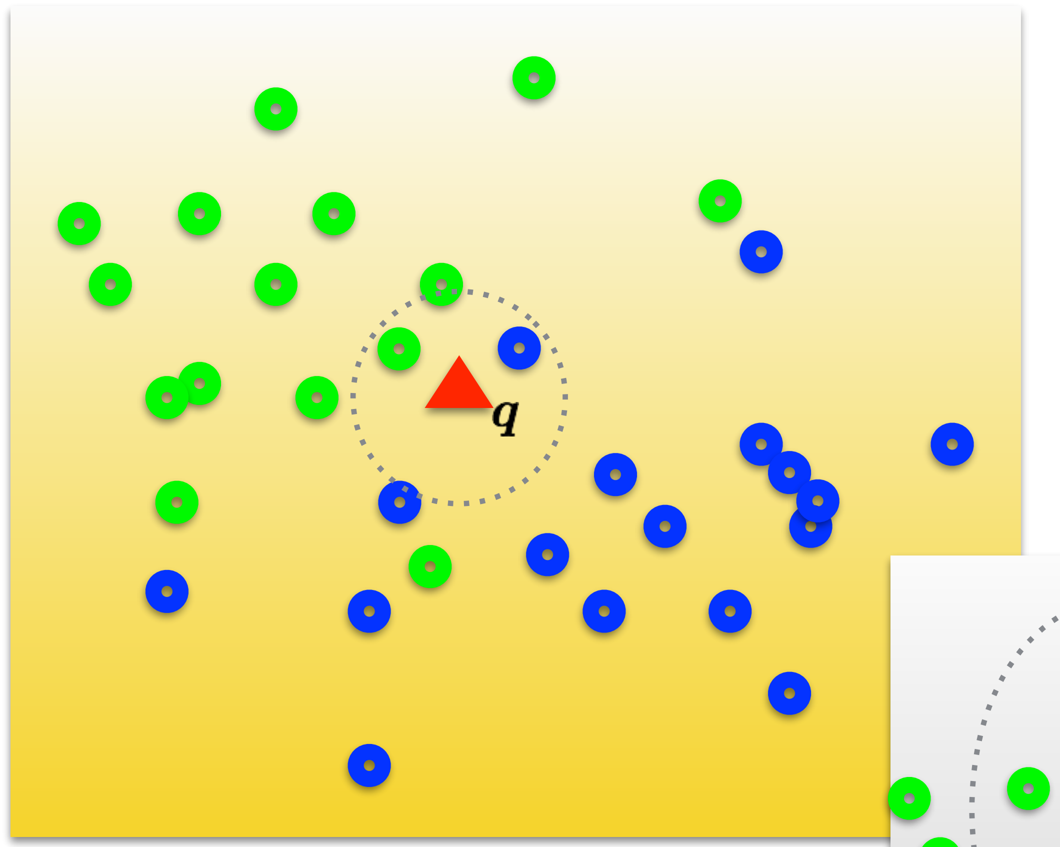
Quantization: image features gets associated to a visual word (nearest cluster center)

Histogram: count the number of visual word occurrences

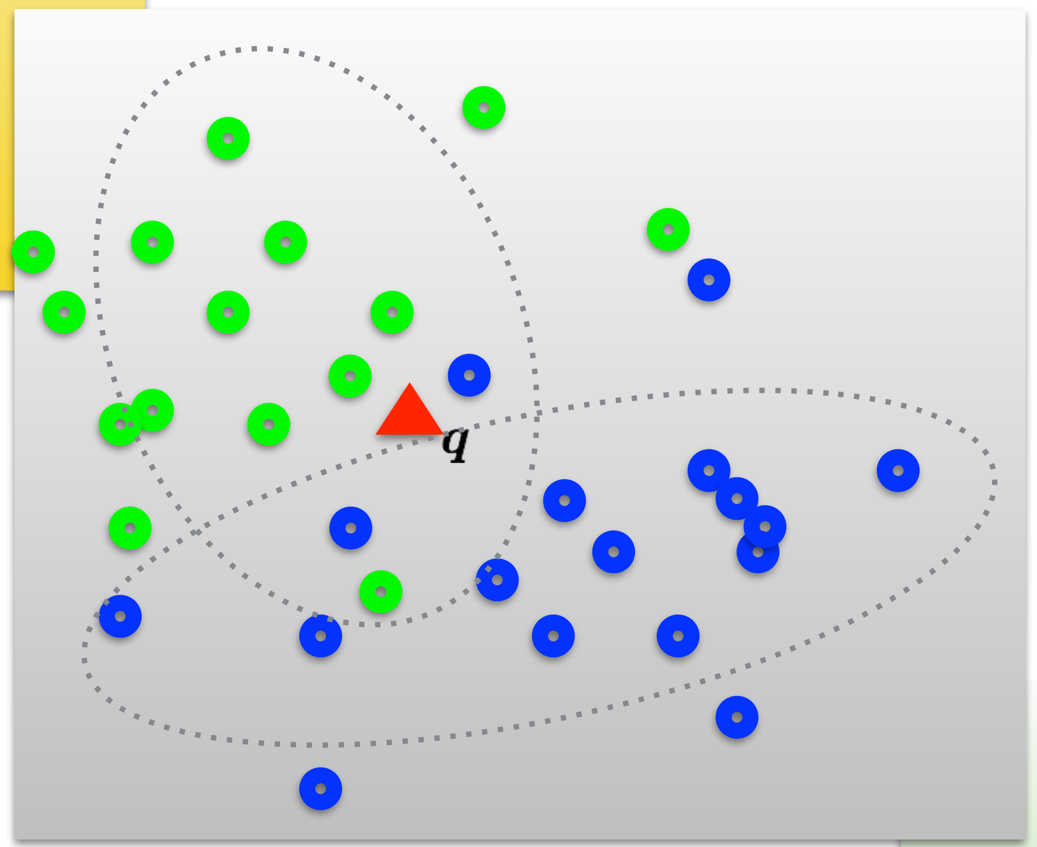


Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 26: Re-cap **Bag-of-Words** Representation

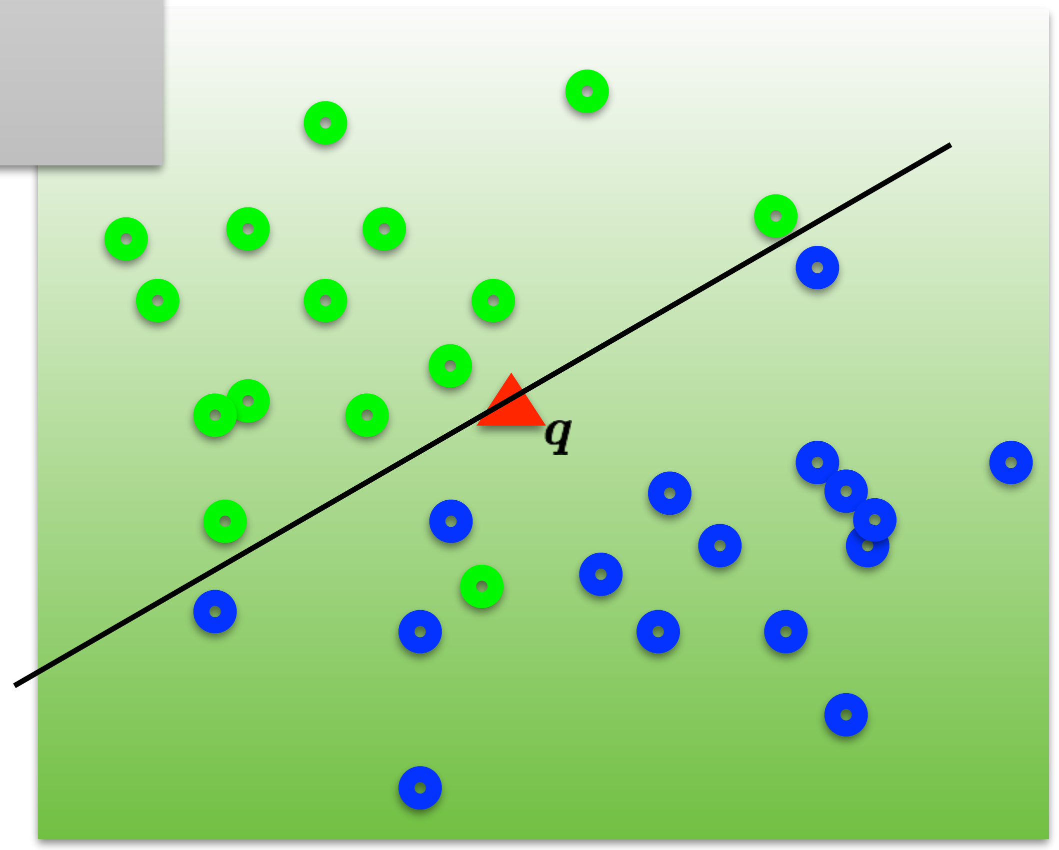


K nearest neighbors



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

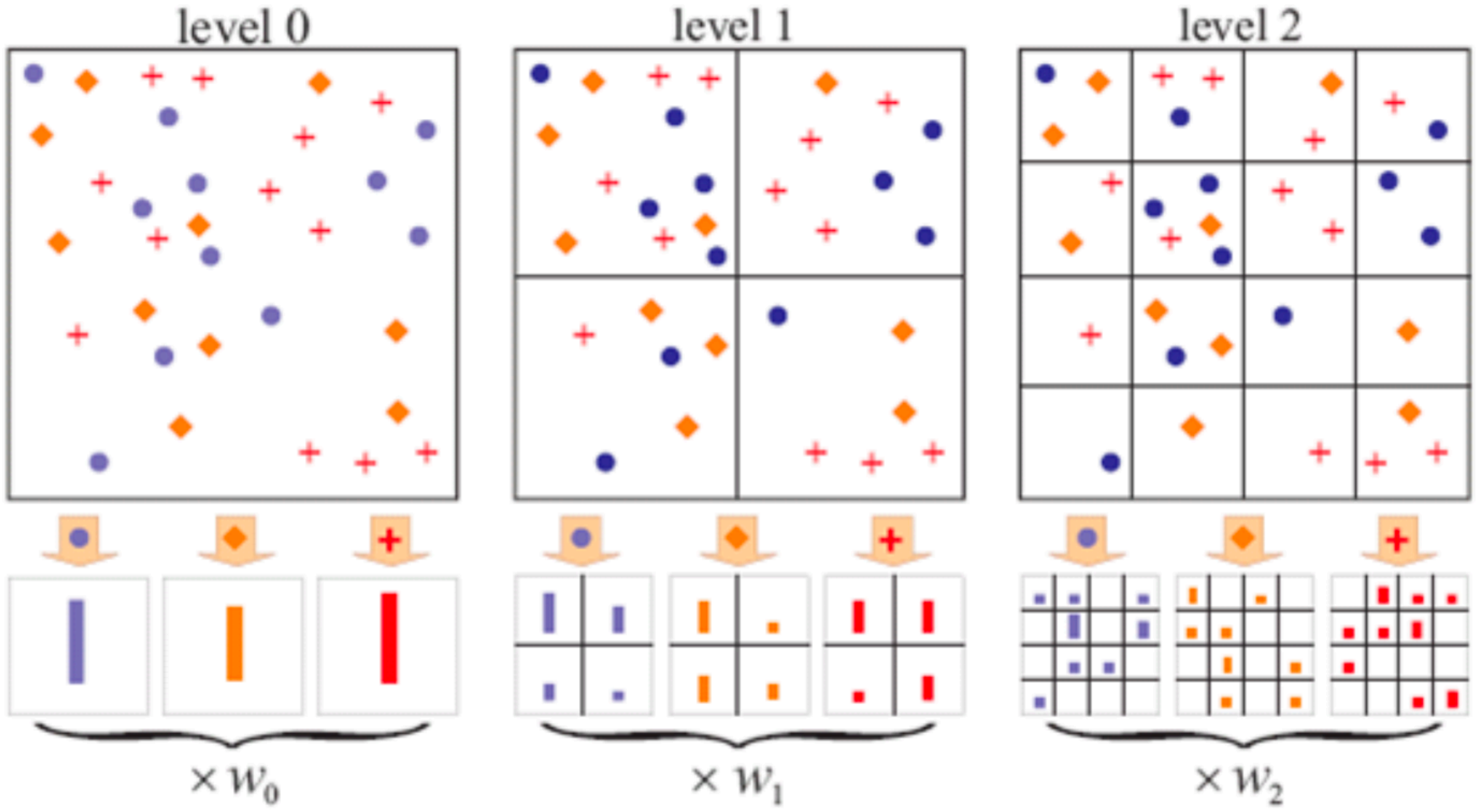


Fig. 16.8 in Forsyth & Ponce (2nd ed.).
Original credit: Lazebnik et al., 2006

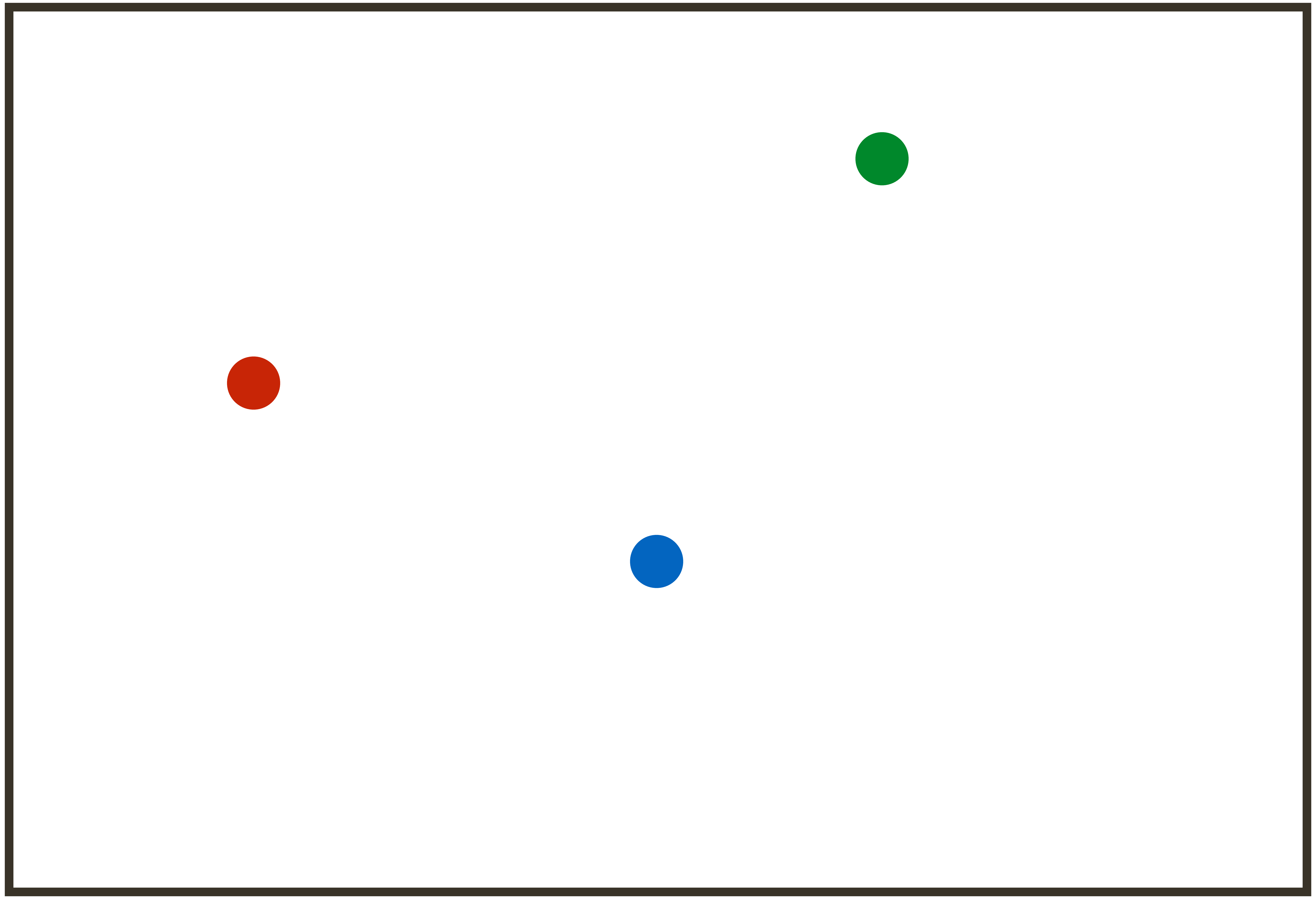
VLAD (Vector of Locally Aggregated Descriptors)

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

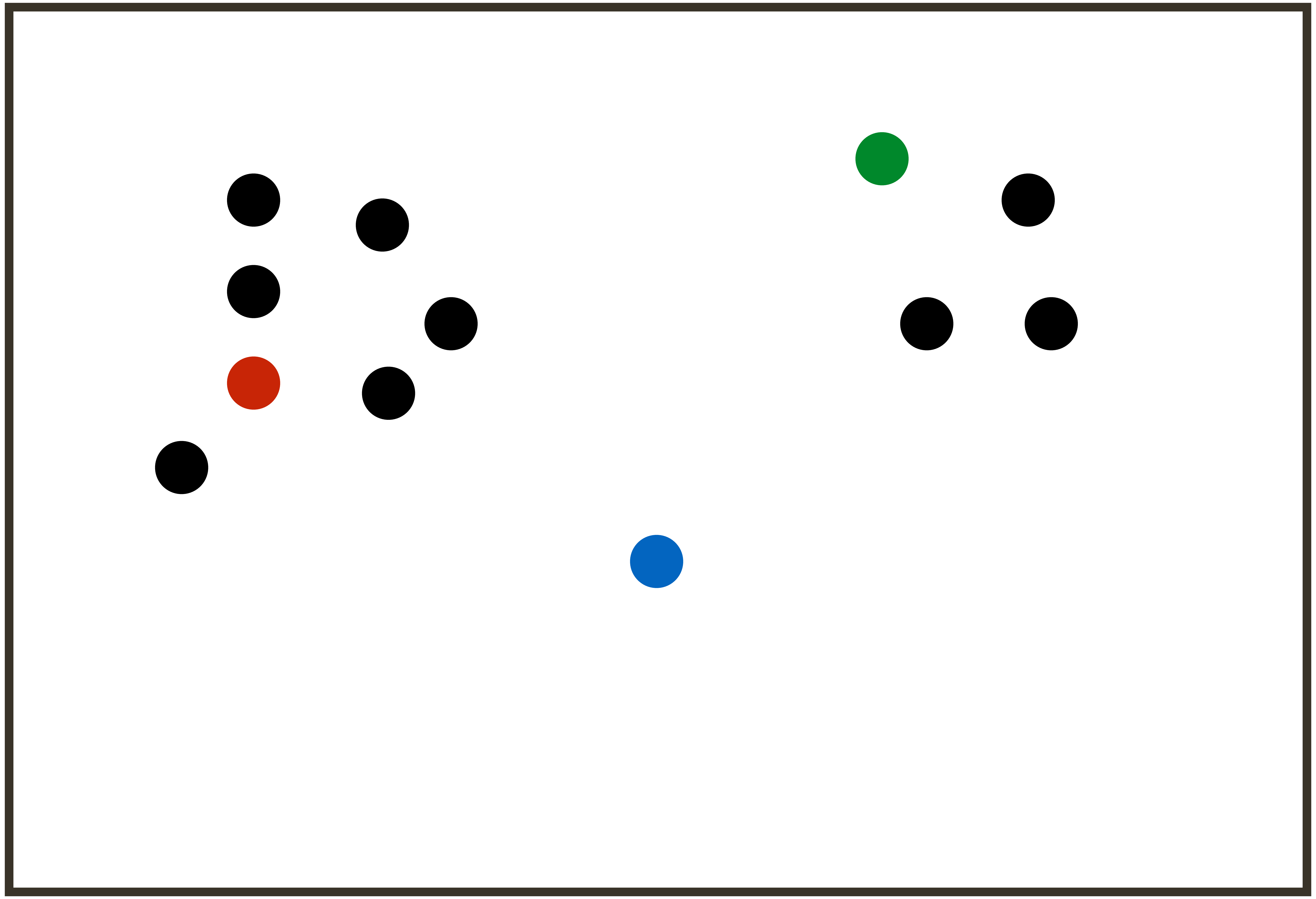
For example, it might be useful to describe how local descriptors are quantized to their visual words

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

Example: VLAD

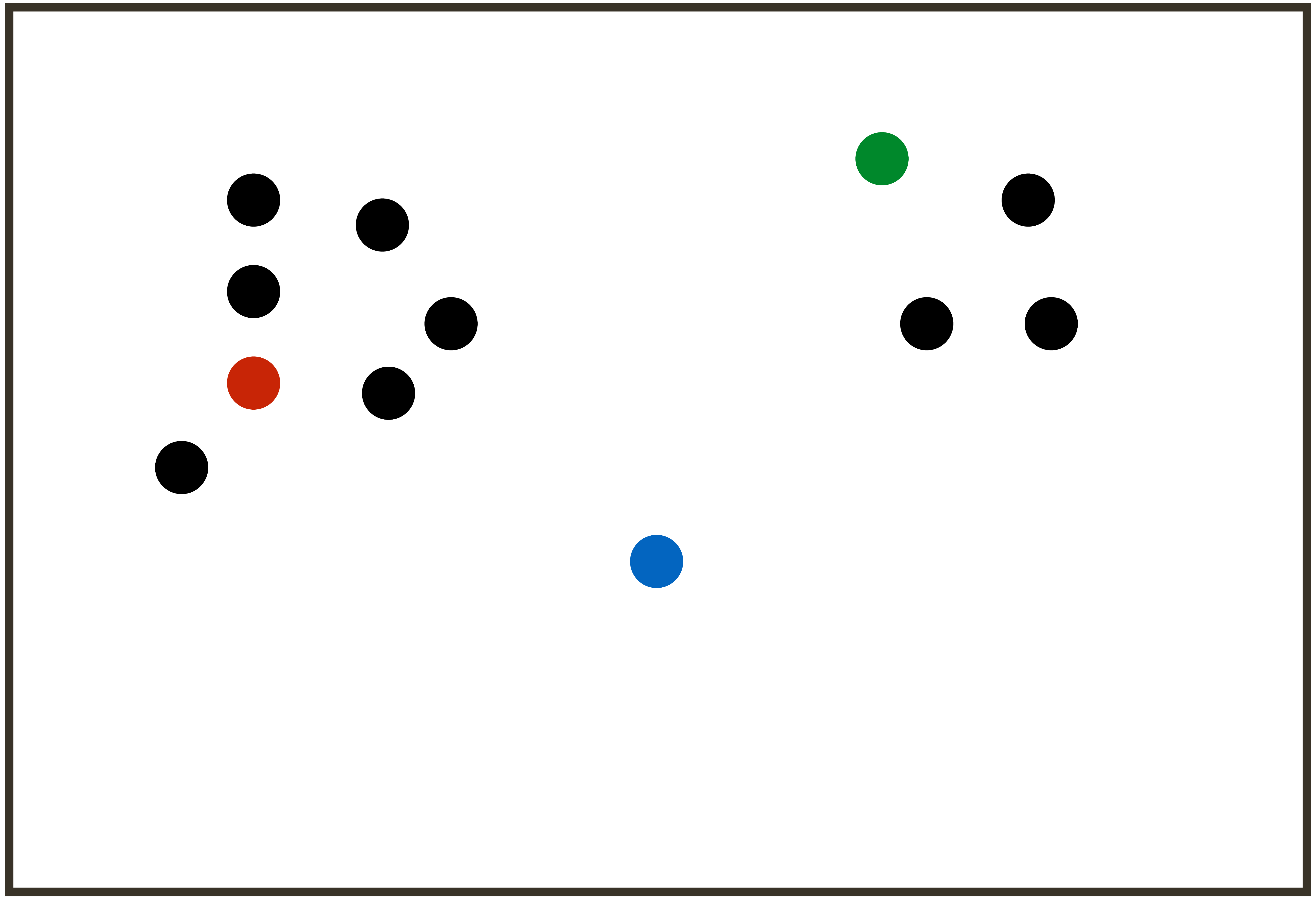


Example: VLAD

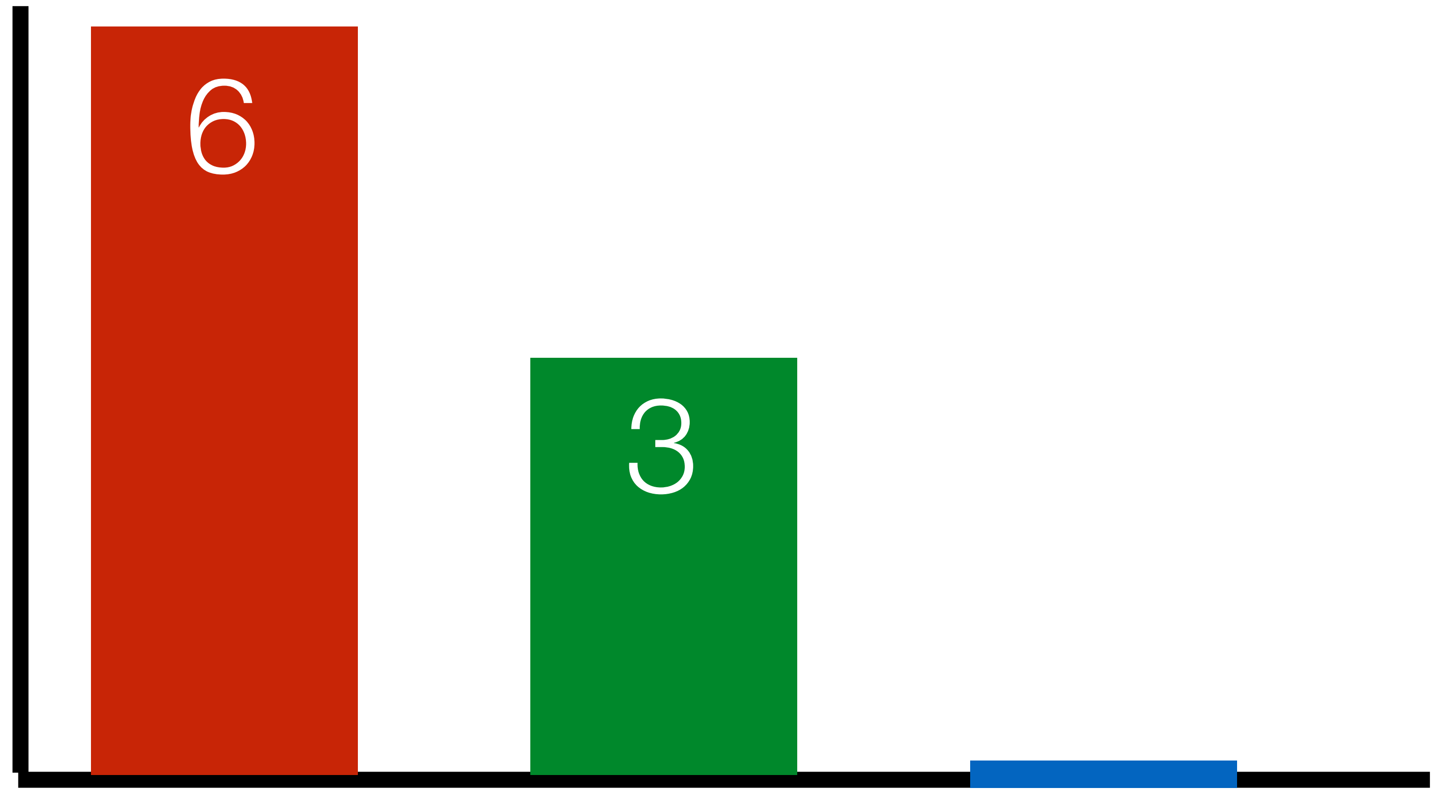


Bag of Word

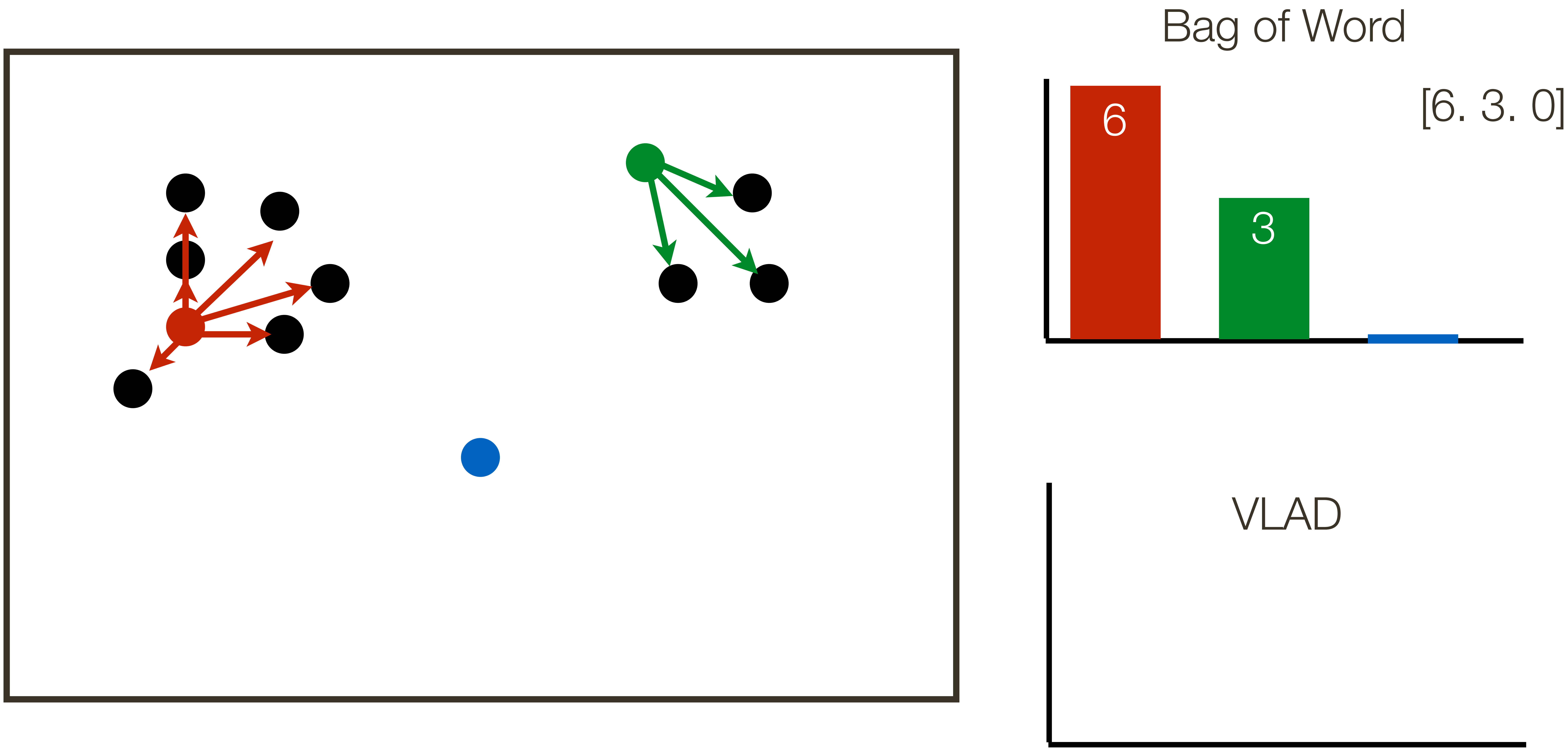
Example: VLAD



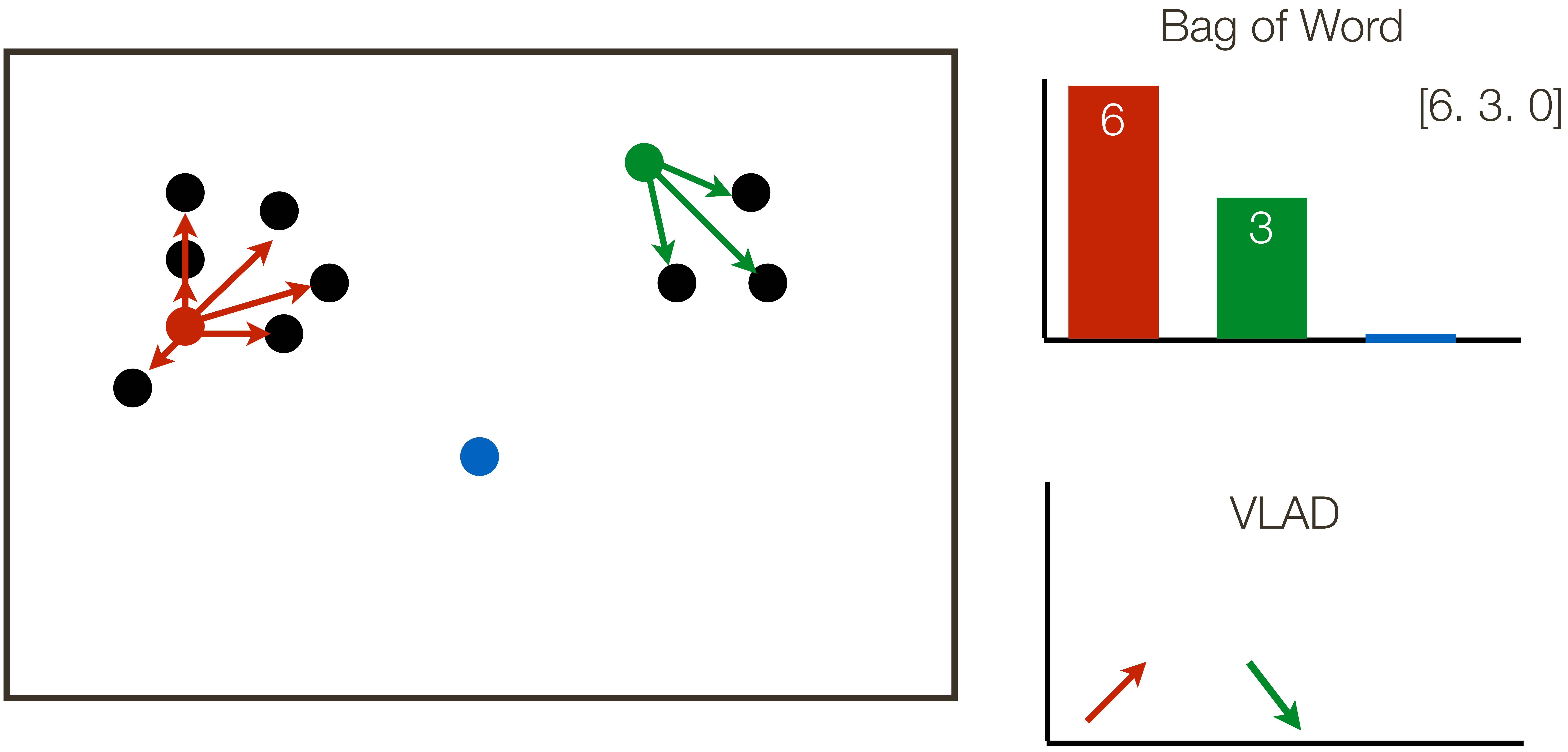
Bag of Word



Example: 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

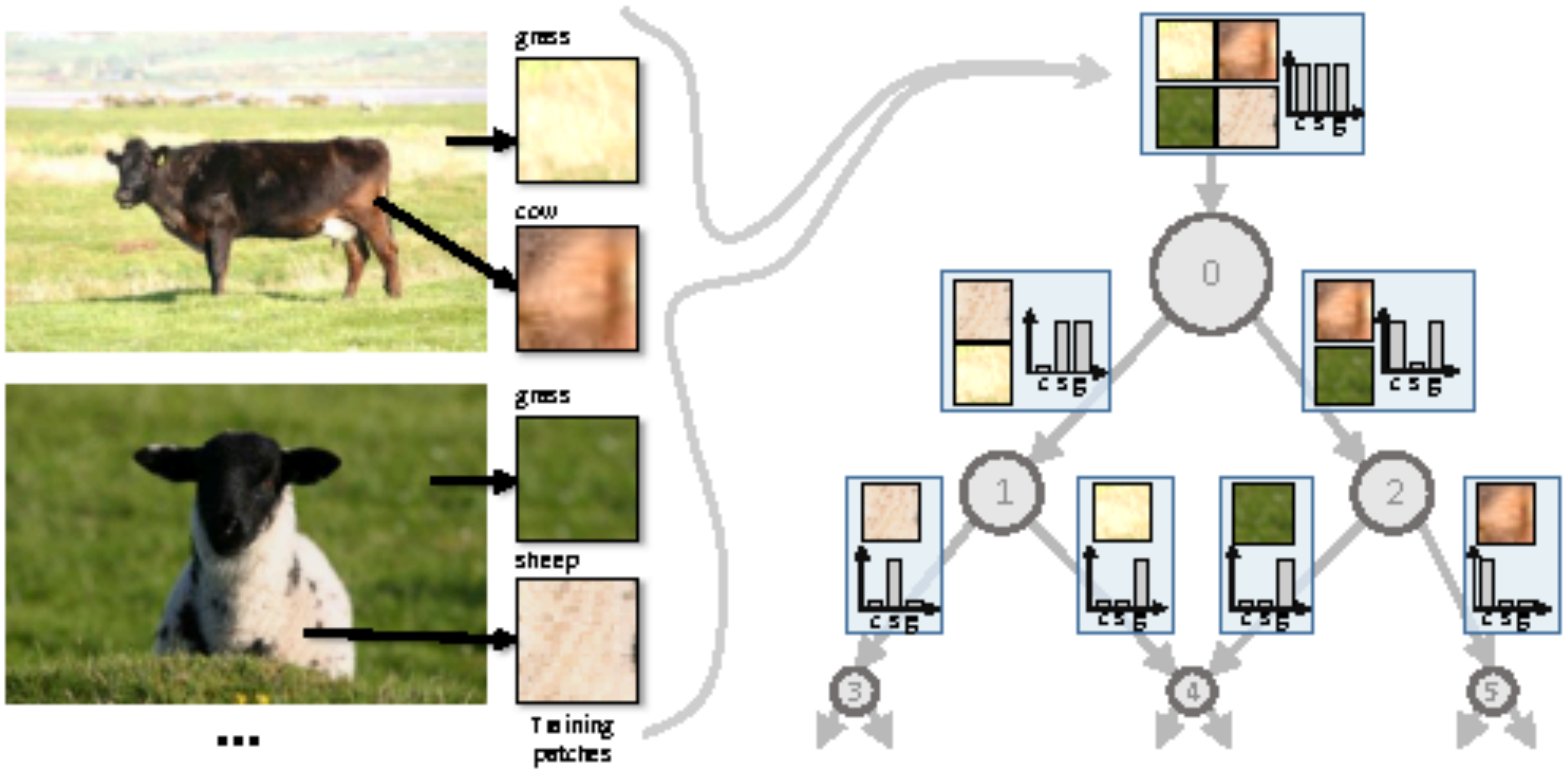
Lecture 25: Forms of Classifiers

Classification strategies fall under two broad types: **parametric** and **non-parametric**.

Parametric classifiers are **model driven**. The parameters of the model are learned from training examples. New data points are classified by the learned model.
— fast, compact
— flexibility and accuracy depend on model assumptions

Non-parametric classifiers are **data driven**. New data points are classified by comparing to the training examples directly. "The data is the model".
— slow
— highly flexible decision boundaries

Decision Tree



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?

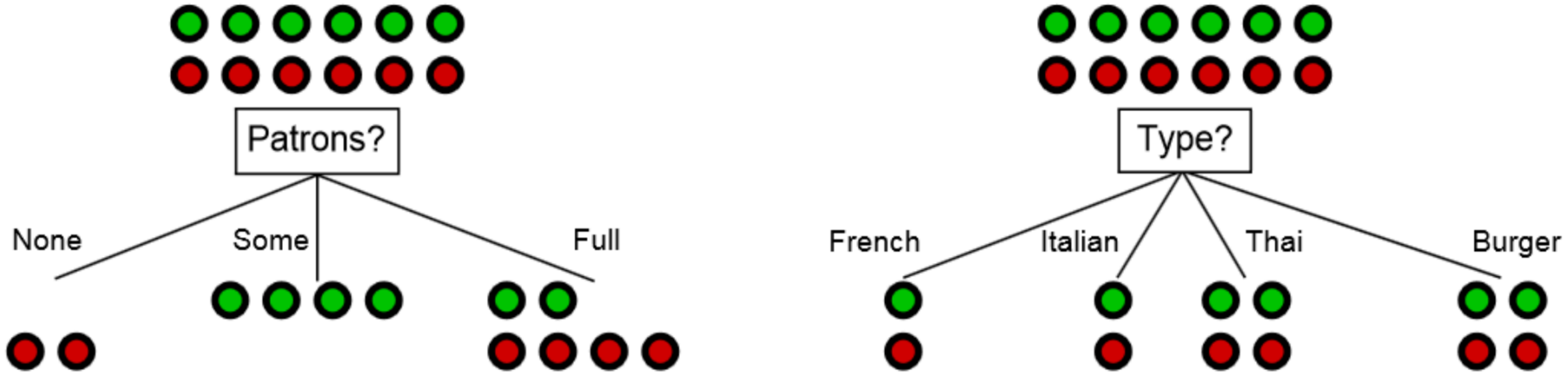


Figure credit: Russell and Norvig (3rd ed.)

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 **information 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:

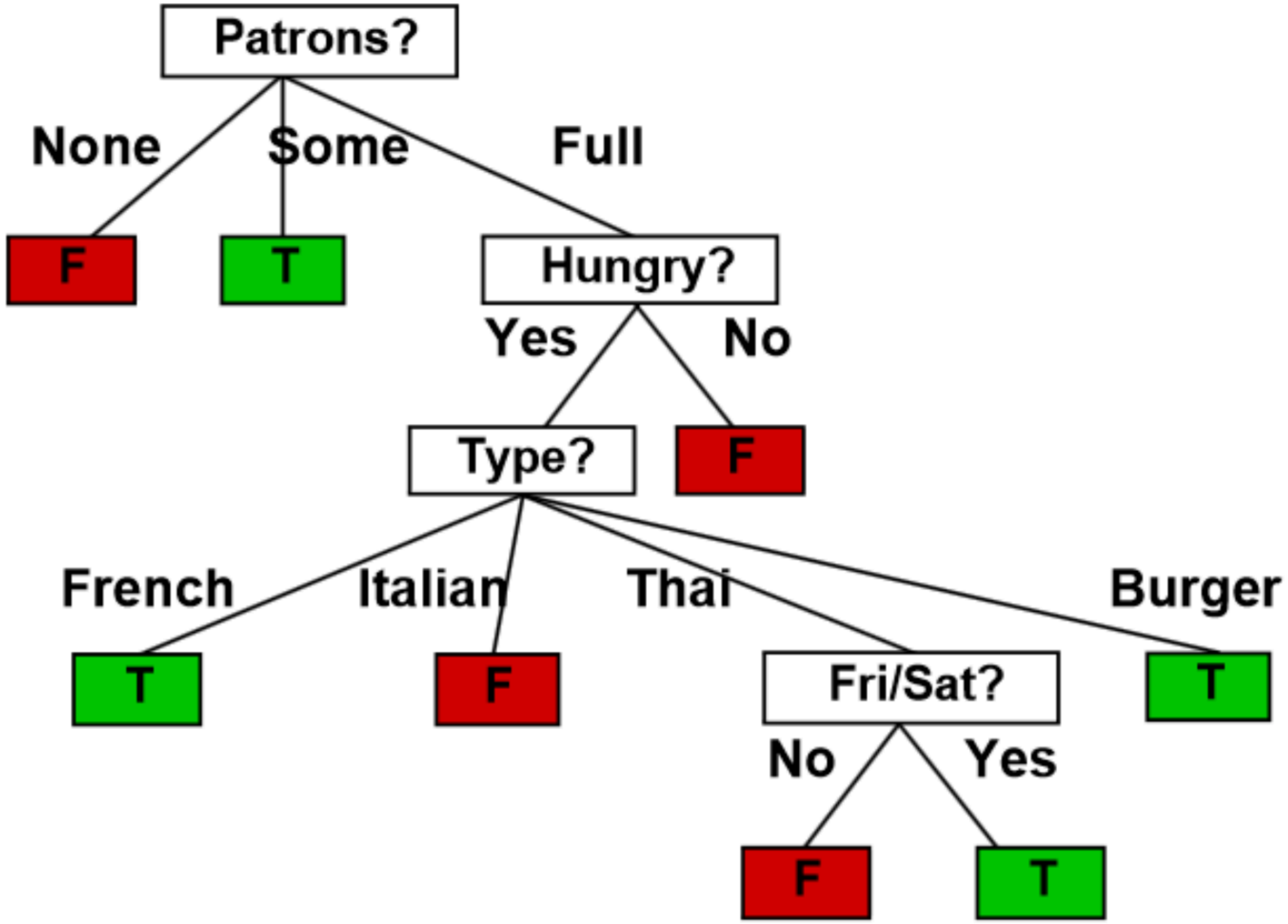


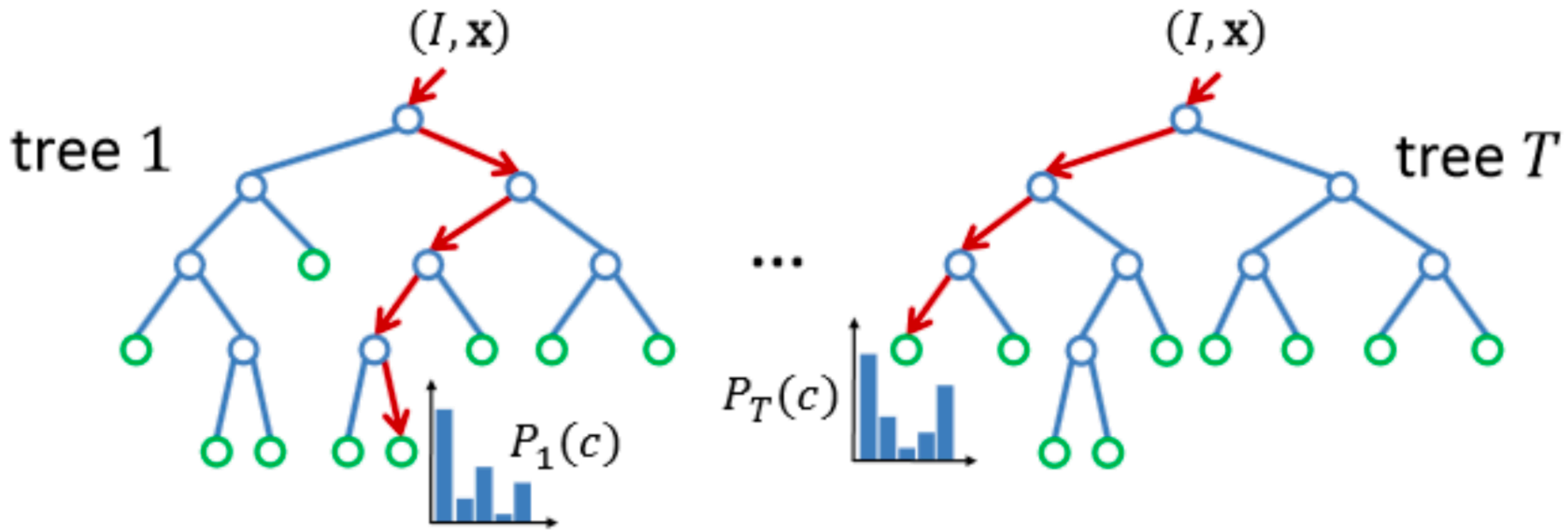
Figure credit: Russell and Norvig (3rd ed.)

Decision Tree

A **random forest** is an ensemble of decision trees.

Randomness is incorporated via training set sampling and/or generation of the candidate binary tests

The prediction of the random forest is obtained by averaging over all decision trees.



Forsyth & Ponce (2nd ed.) Figure 14.19. Original credit: J. Shotton et al., 2011

Example 1: Kinect

Kinect allows users of Microsoft's Xbox 360 console to interact with games using natural body motions instead of a traditional handheld controller. The pose (joint positions) of the user is predicted using a random forest trained on depth features.

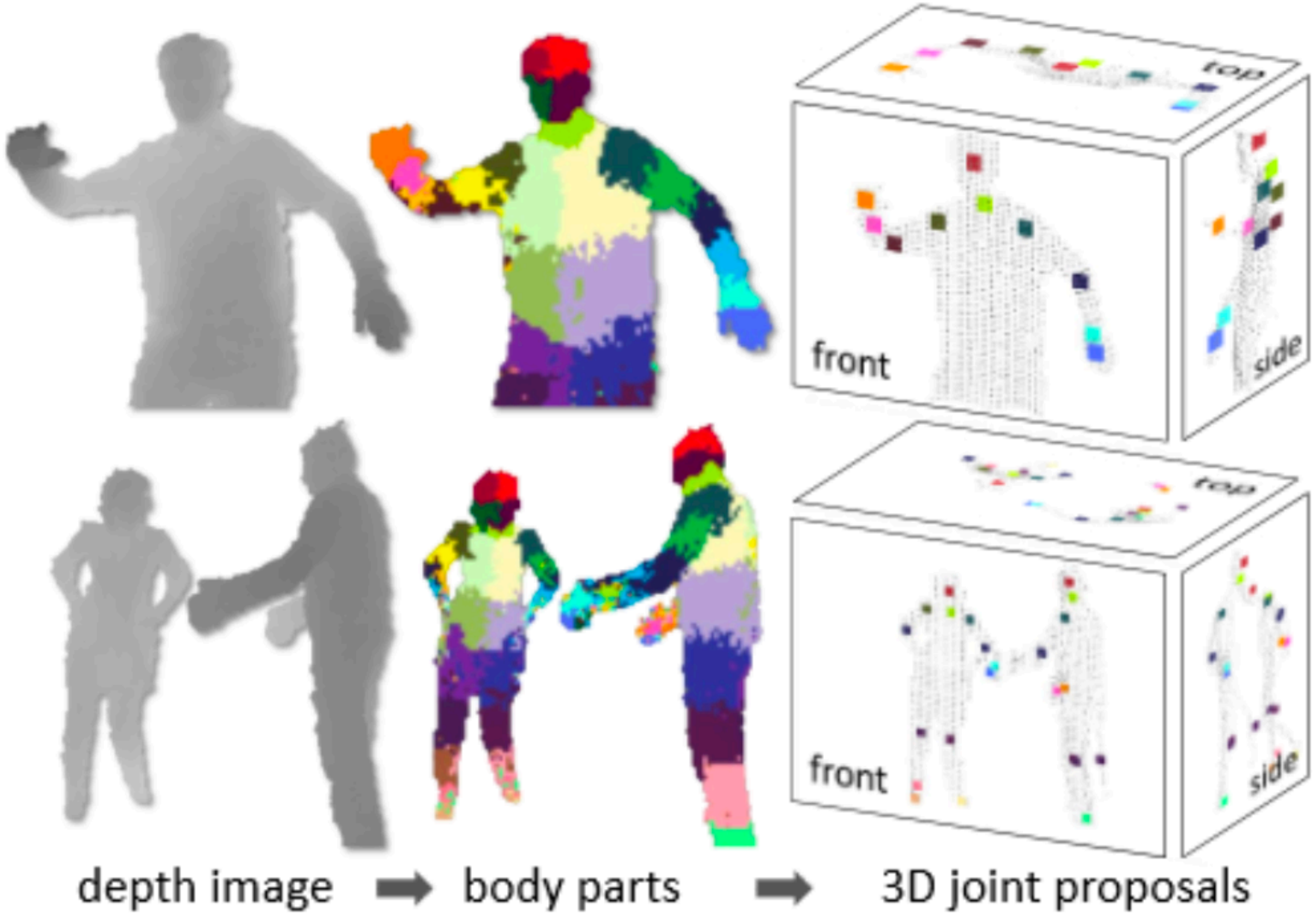
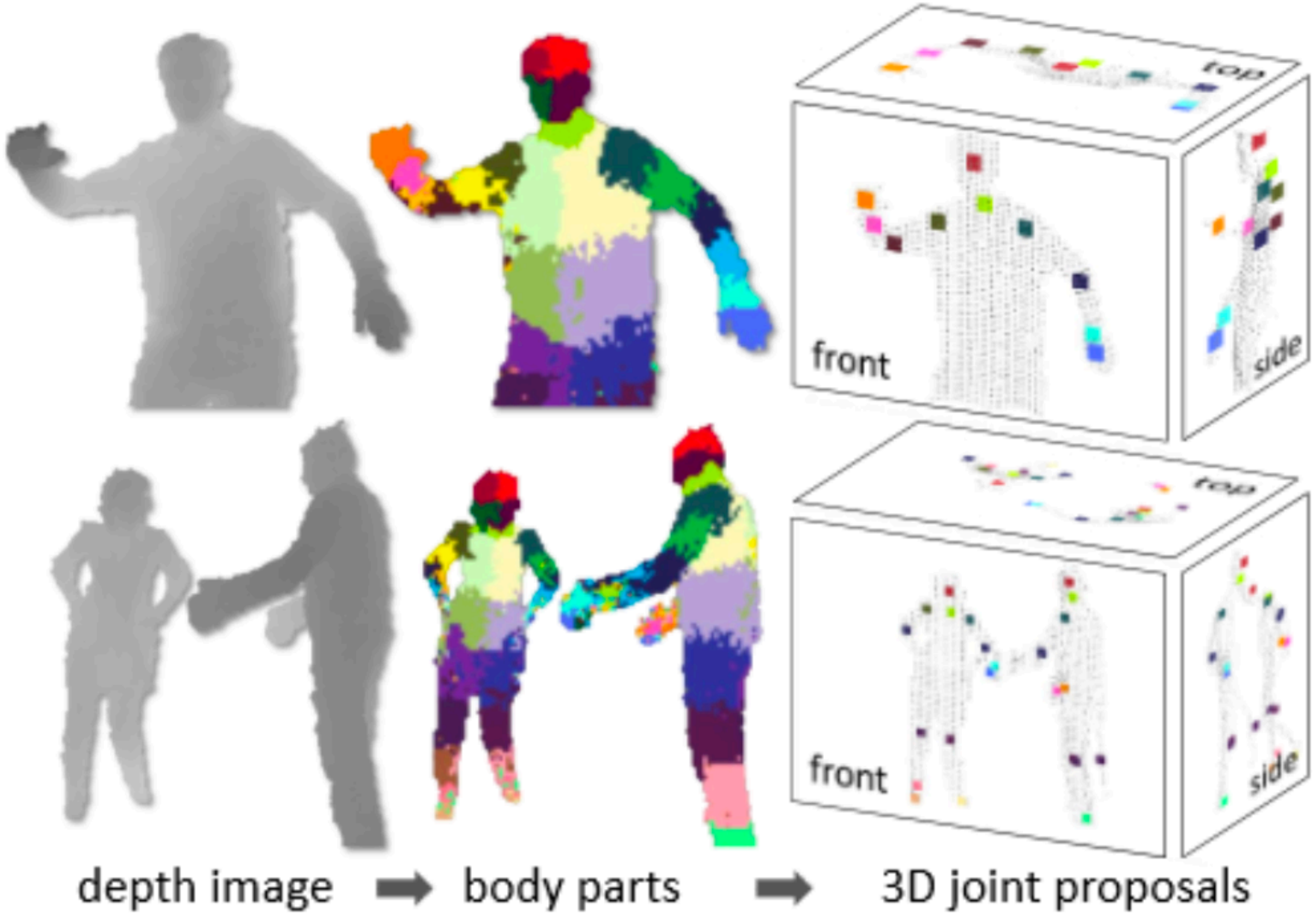


Figure credit: J. Shotton et al., 2011

Example 1: Kinect

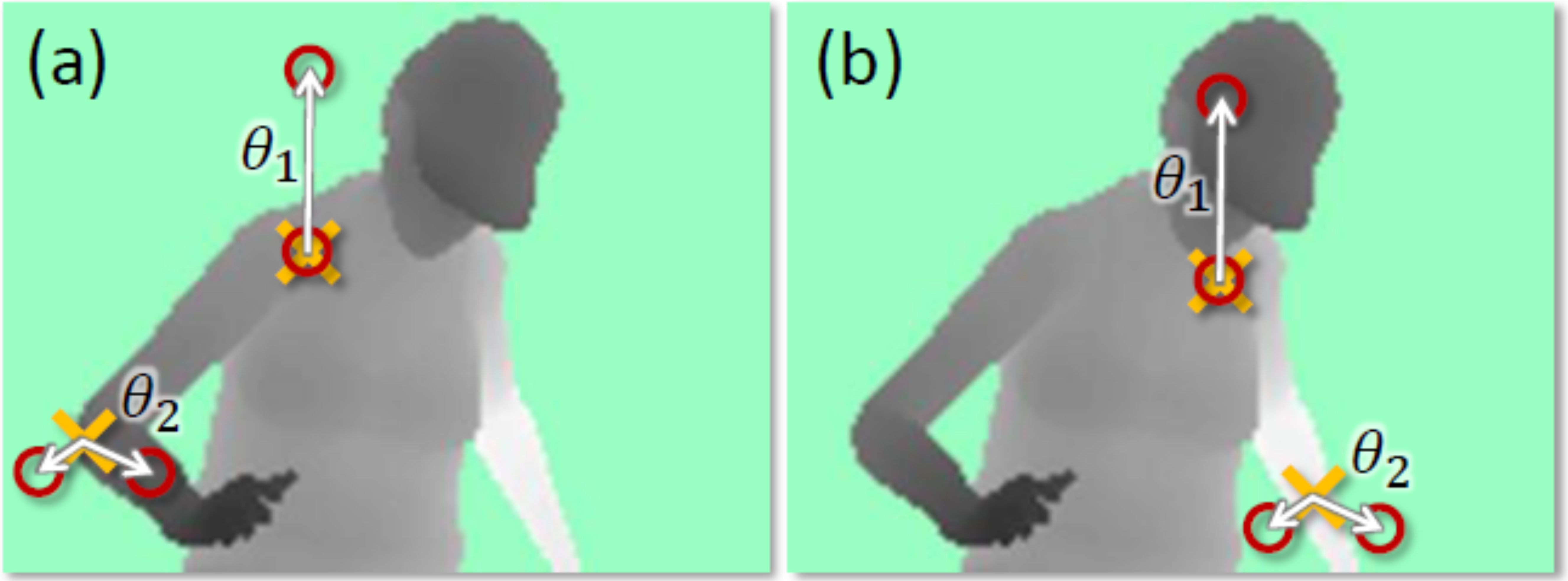
Kinect allows users of Microsoft's Xbox 360 console to interact with games using natural body motions instead of a traditional handheld controller. The pose (joint positions) of the user is predicted using a random forest trained on depth features.



Jamie Shotton

Figure credit: J. Shotton et al., 2011

Example 1: Kinect



$$f_\theta(I, \mathbf{x}) = d_I\left(\mathbf{x} + \frac{\mathbf{u}}{d_I(\mathbf{x})}\right) - d_I\left(\mathbf{x} + \frac{\mathbf{v}}{d_I(\mathbf{x})}\right)$$

Figure credit: J. Shotton et al., 2011

Example 1: Kinect

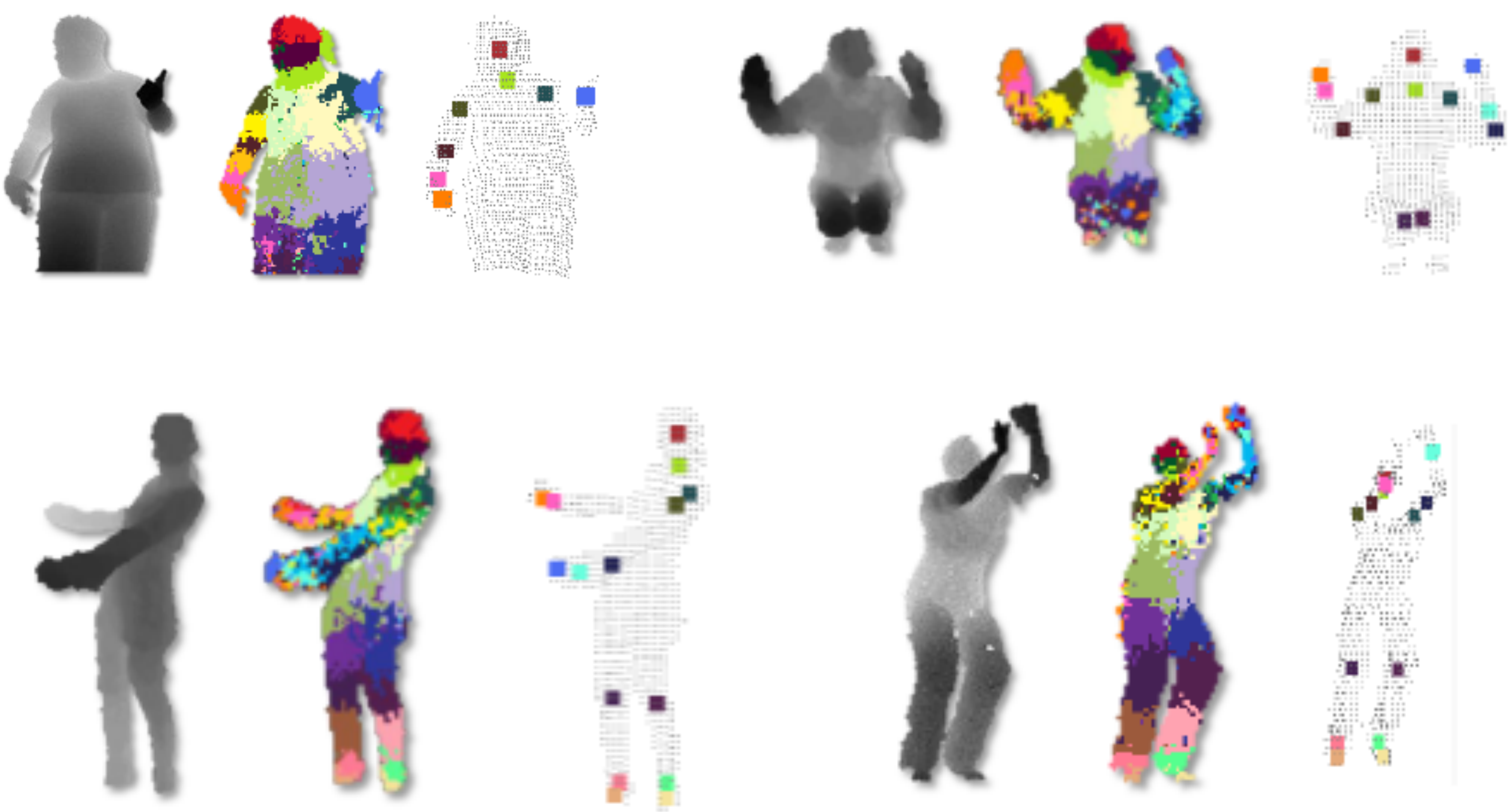


Figure credit: J. Shotton et al., 2011