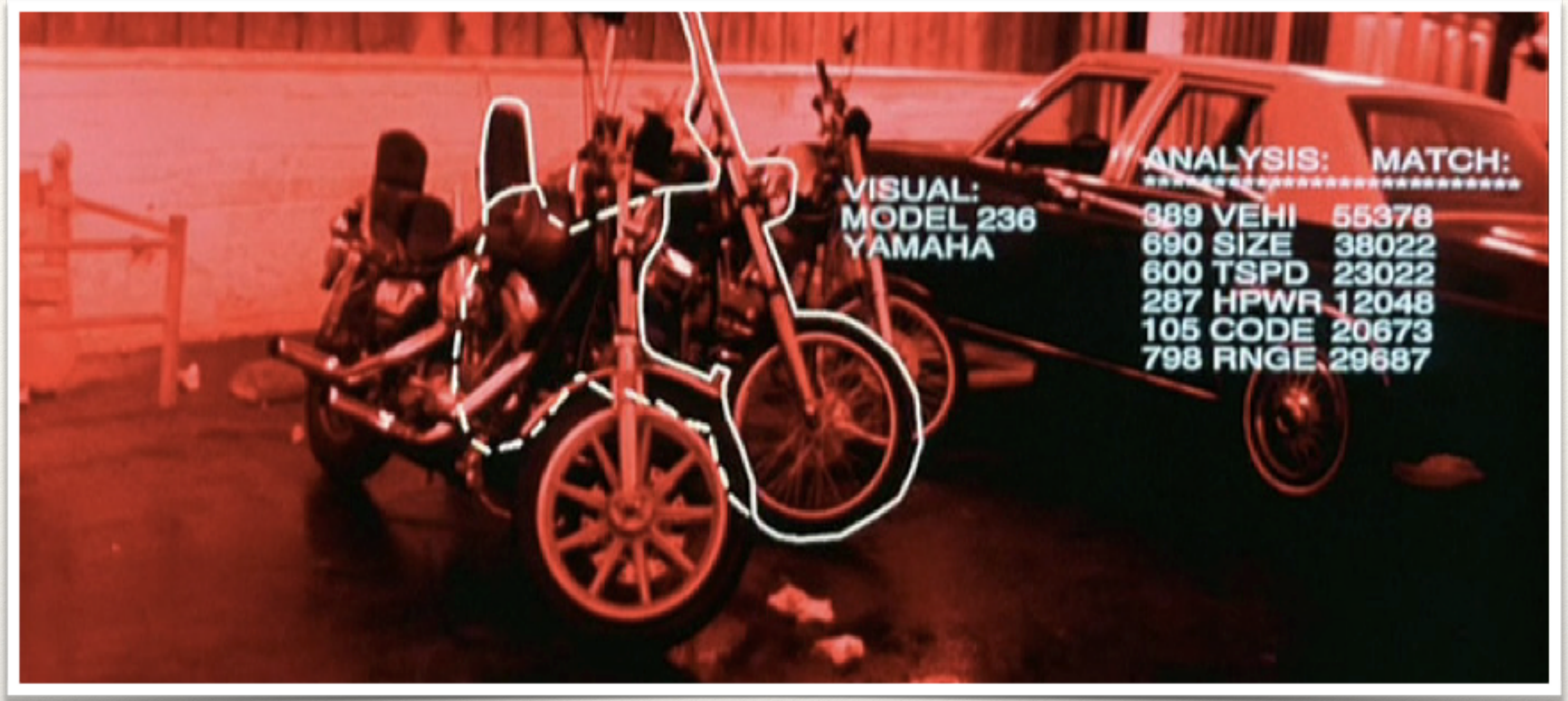
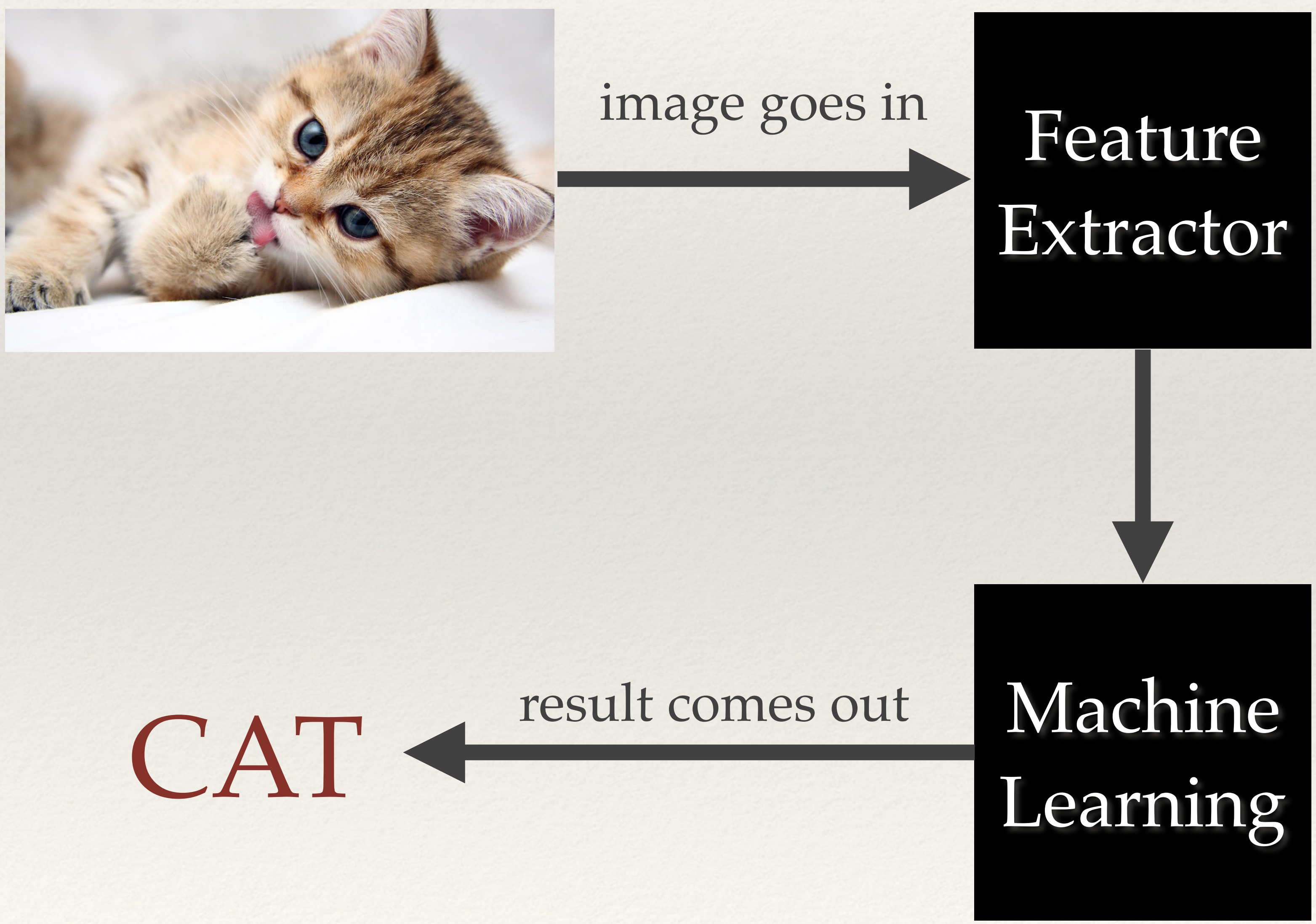


COMP3204/COMP6223: Computer Vision

Image classification and auto-annotation

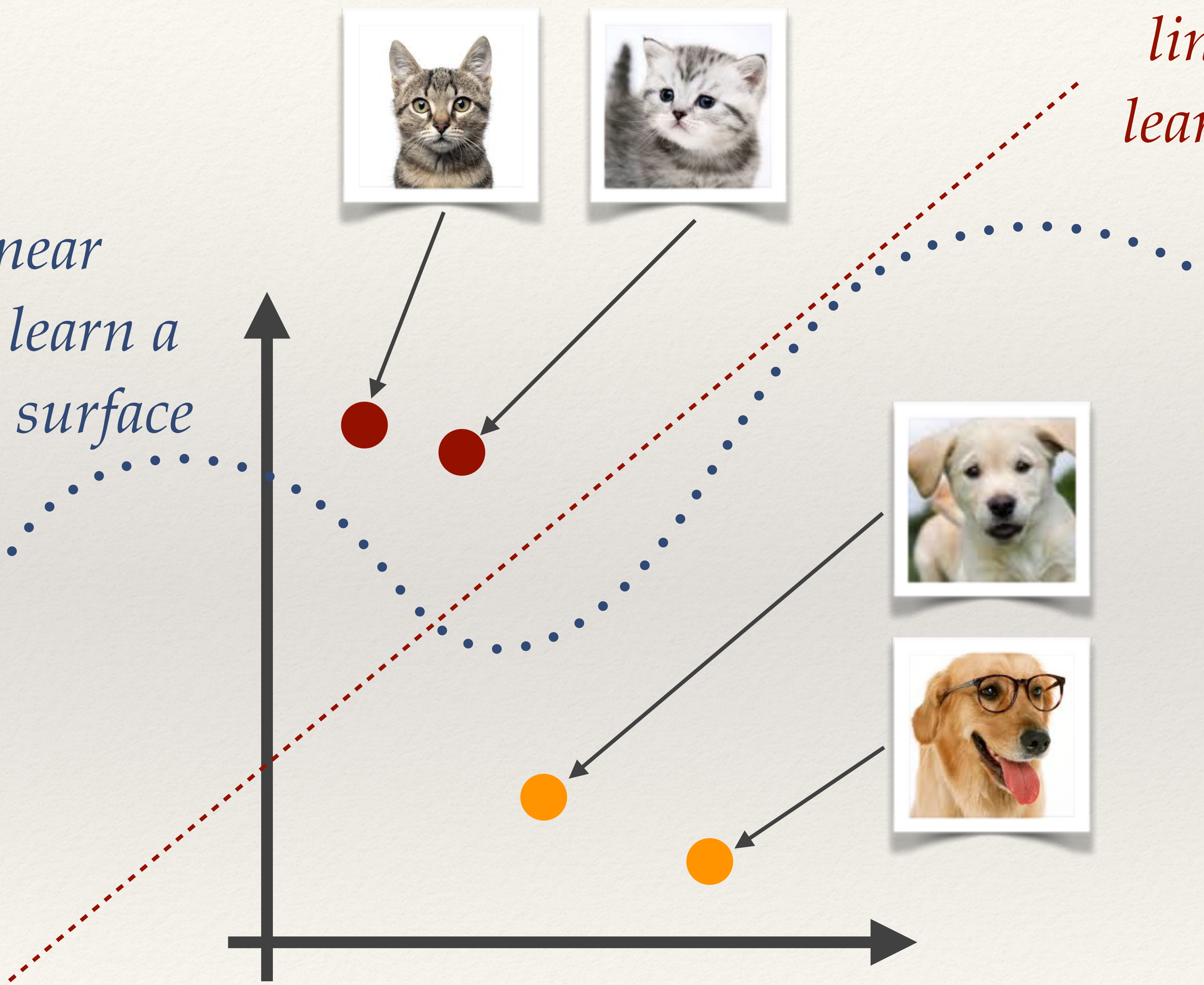
Jonathon Hare
jsh2@ecs.soton.ac.uk

Recap: Computer Vision Systems



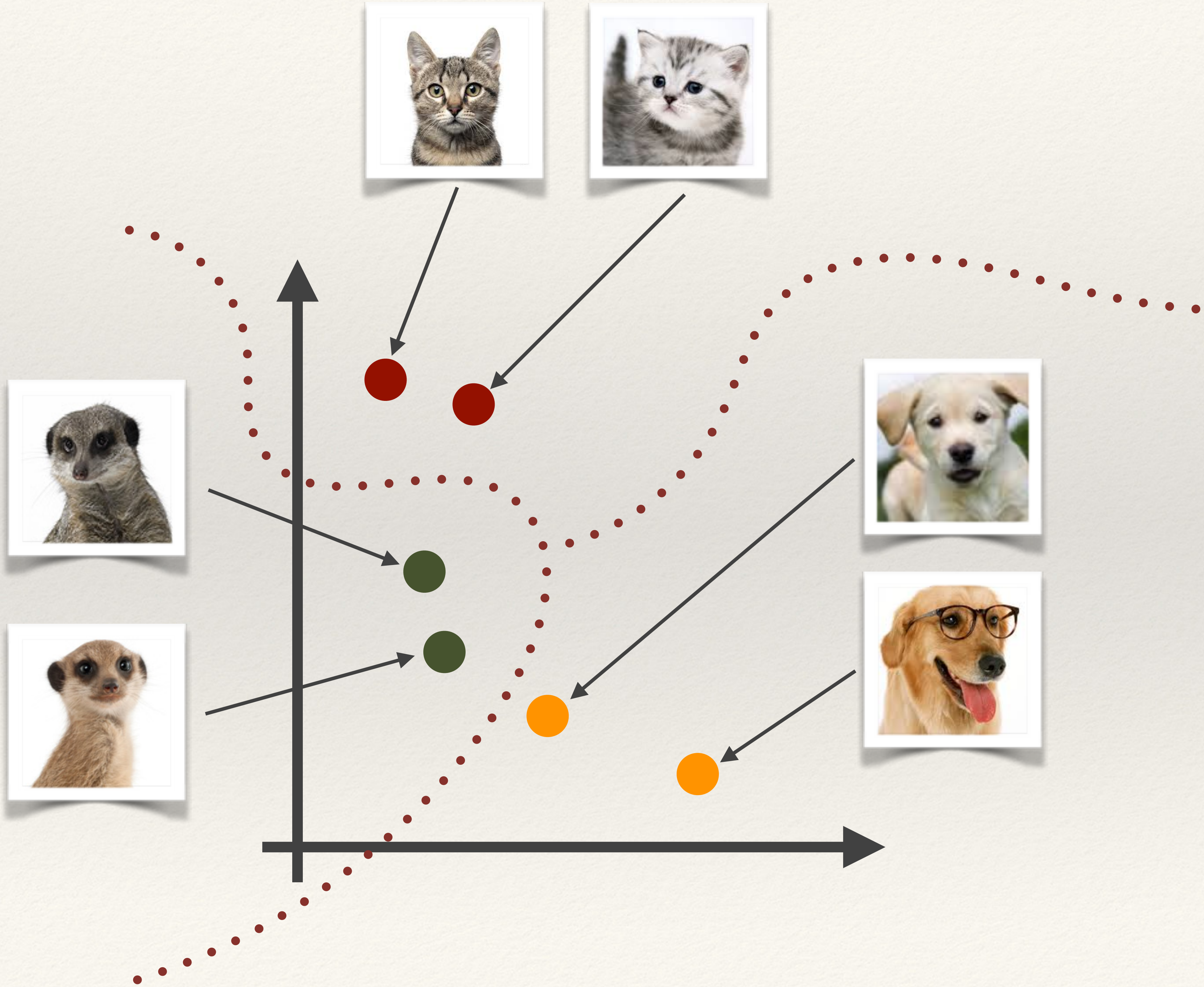
Recap: Binary classification

Non linear classifiers learn a separating surface



linear classifiers learn a separating hyperplane

Recap: Multi-class classification



Multilabel classification

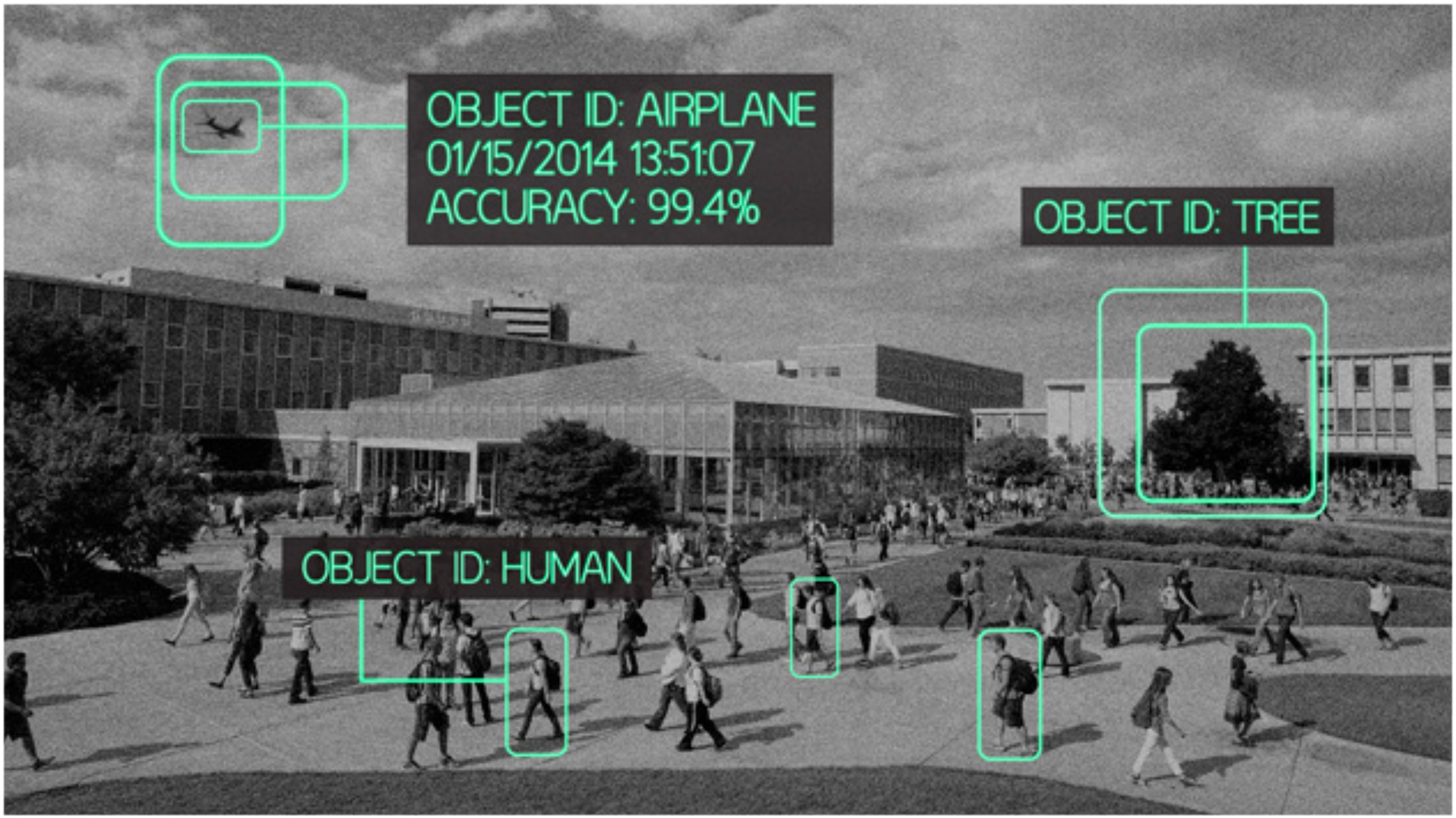
CAT **DOG**

in the context of images often called *Automatic Annotation*

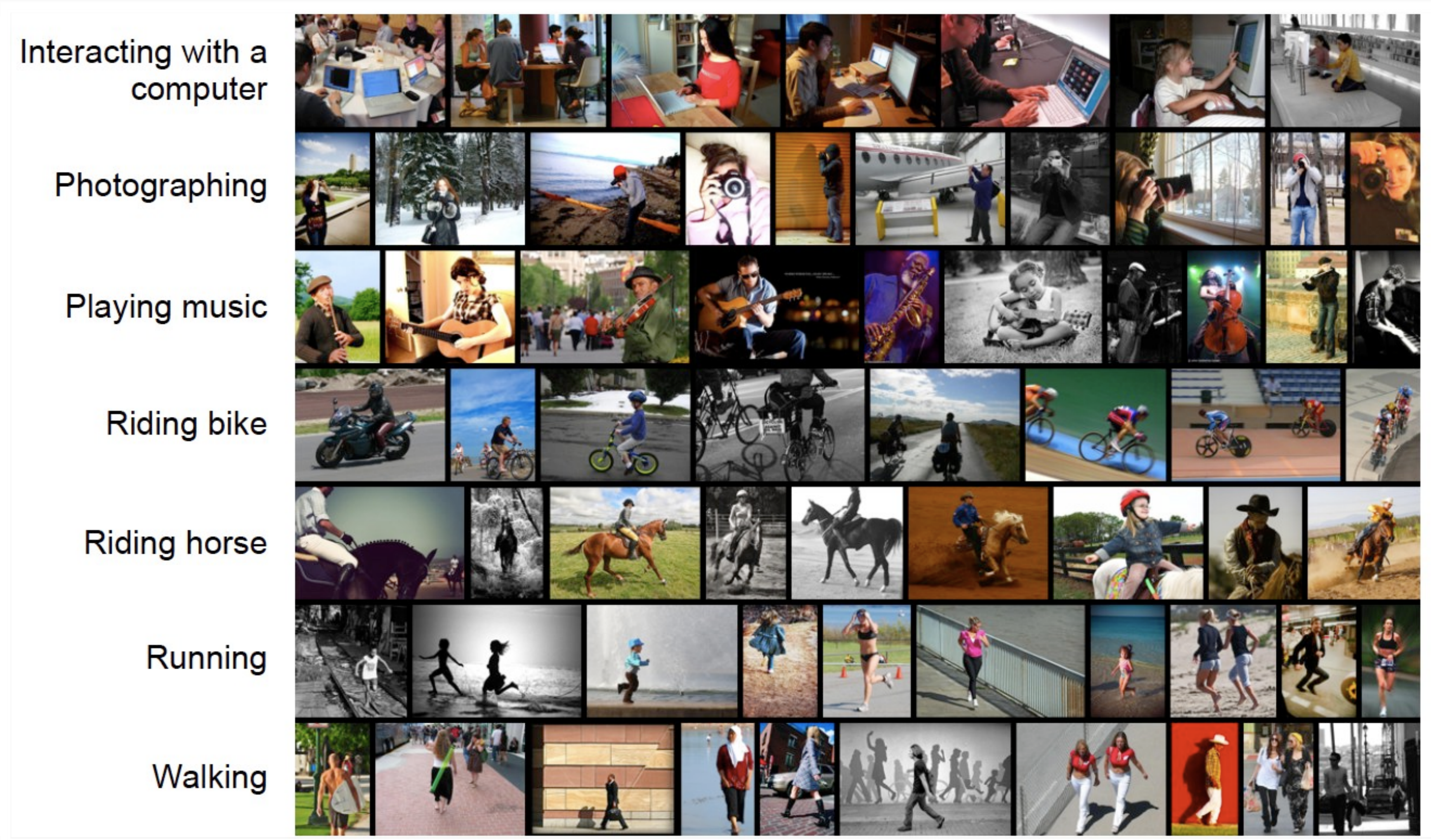
Object Detection/Localisation

Challenges in Computer Vision

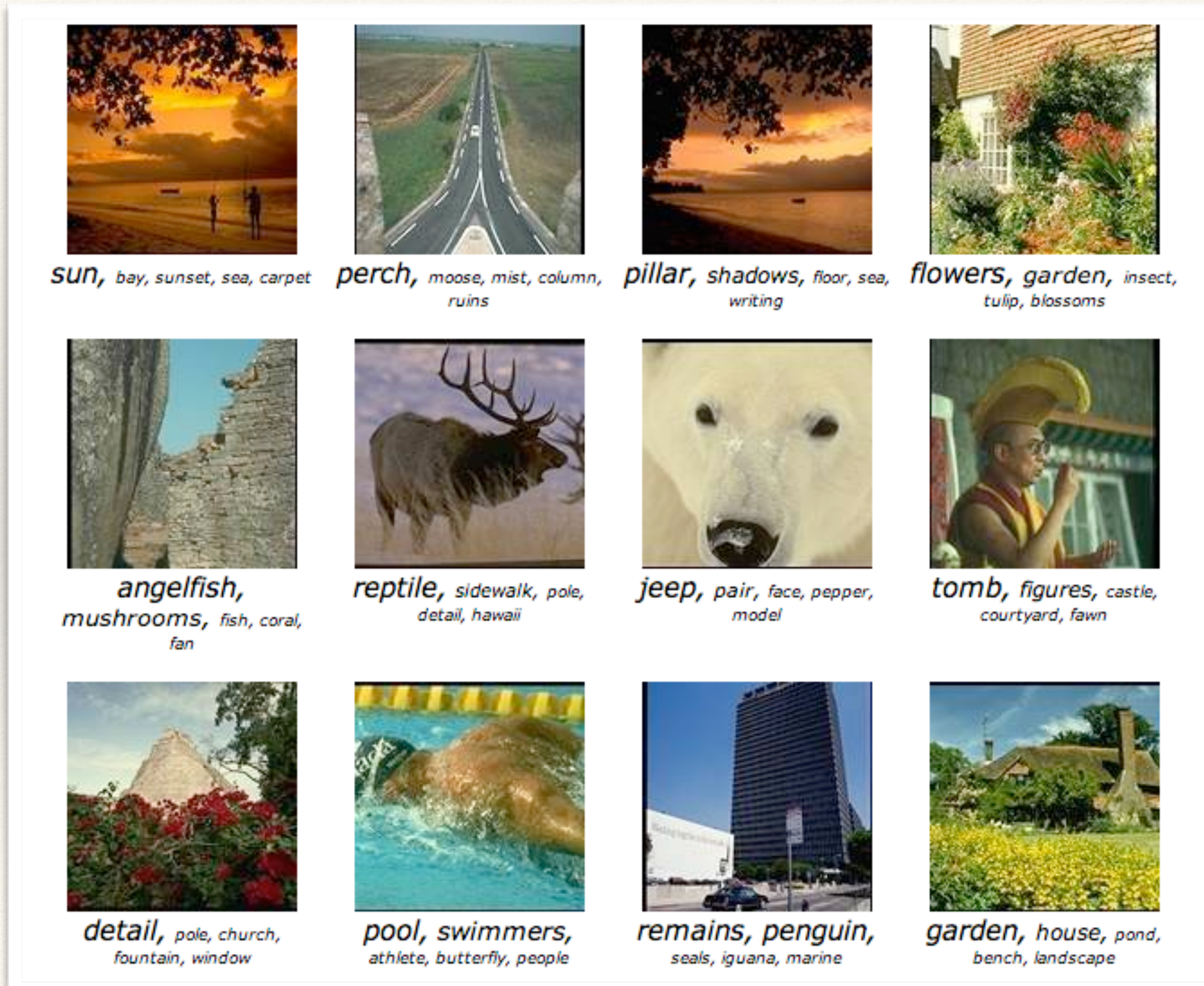
Object Recognition in natural scenes



Scene/Activity Classification



Automatic Annotation



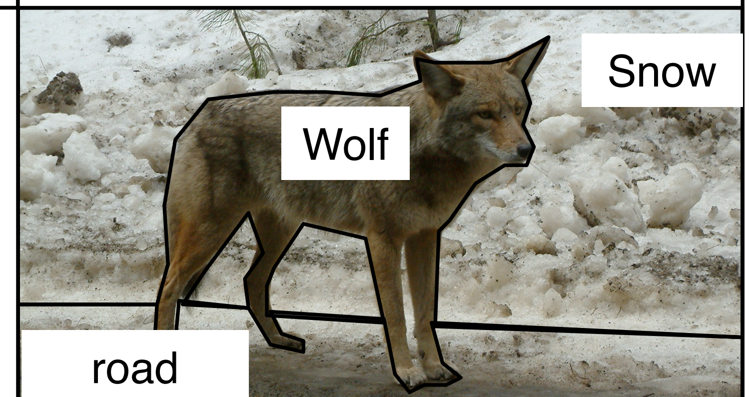
The fundamental problem of computer vision:

The Semantic Gap

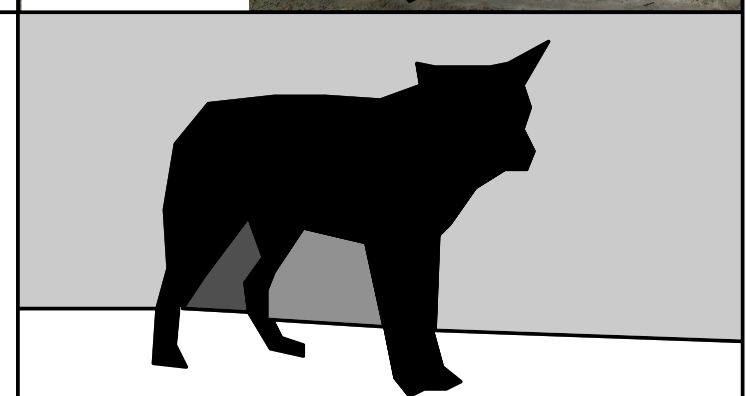
Semantics
object relationships and more

Wolf **on** Road **with** Snow **on Roadside in Yosemite National Park, California on 24/1/2004 at 23:19:11GMT**

Object Labels
symbolic names of objects



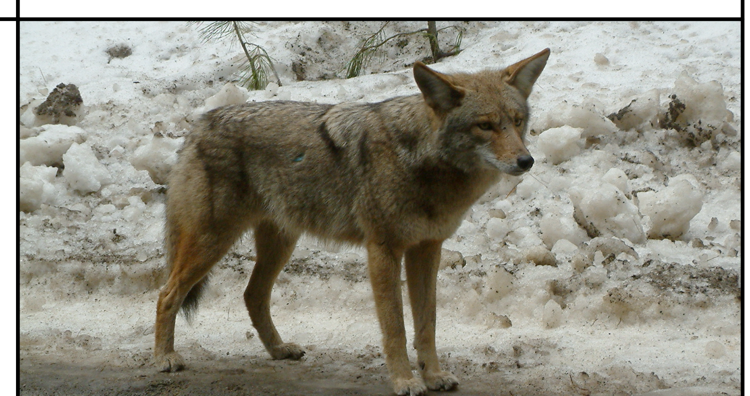
Objects
prototypical combinations of descriptors



Descriptors
feature-vectors

Segmented blobs, Salient regions, Pixel-level histograms, Fourier descriptors, etc...

Raw Media
images



A car parked on double yellow lines

A potted history

Object Recognition

- 1999 - SIFT matching
  - Very powerful, but computationally demanding
- 2001 - Cascades of Haar-like features
  - Very popular for face detection
- 2006 - SURF matching
  - Combined ideas from SIFT and the integral images used for computing Haar-like features

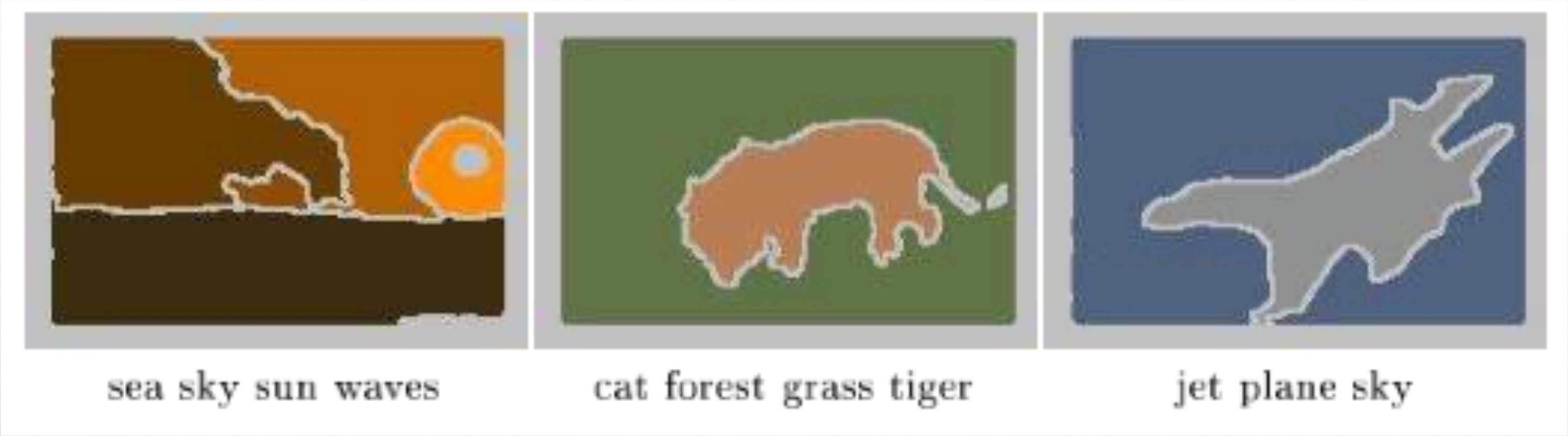
Interest in auto-annotation grew from the late 90s

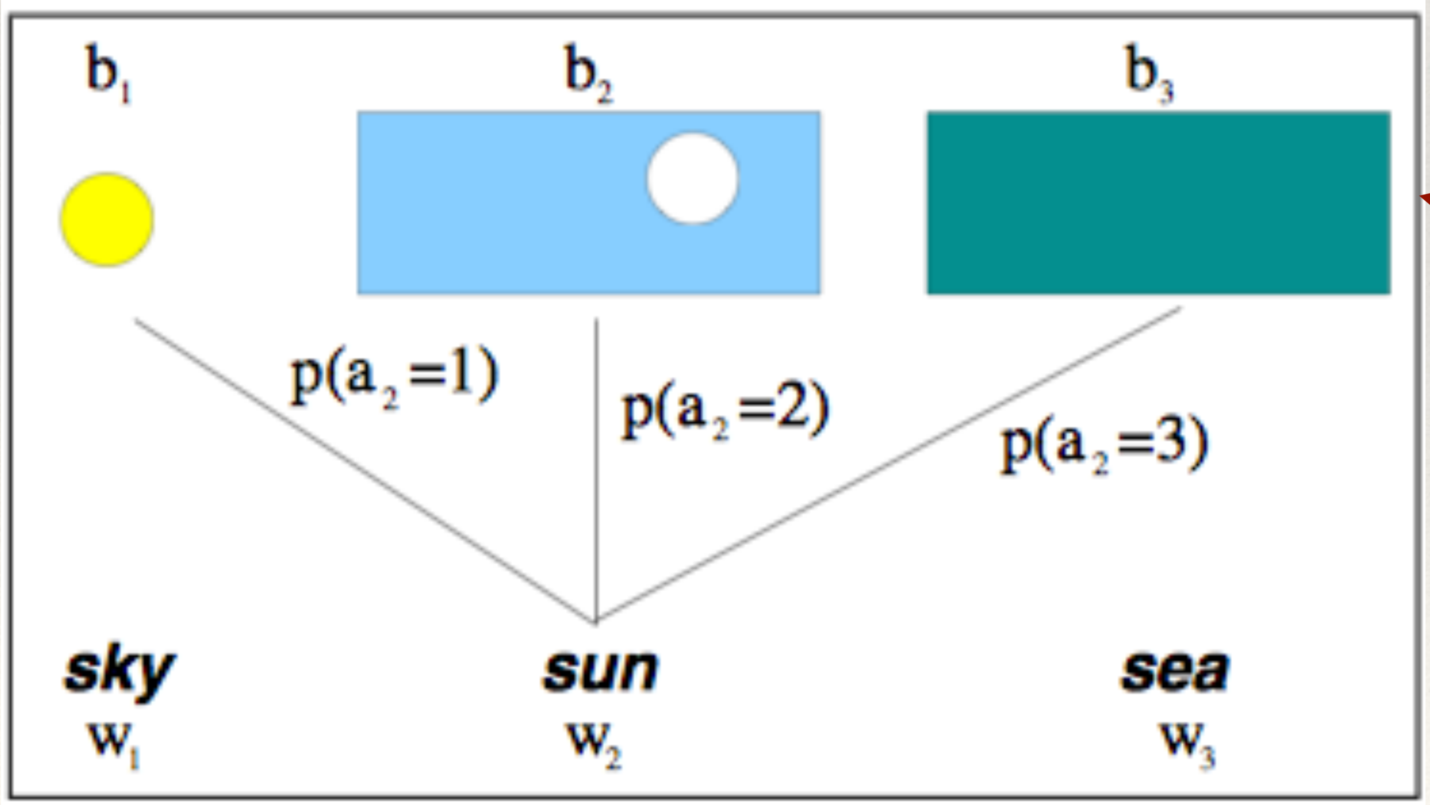
Bags of "Visual Words" were rather important!

Aside: Optimal codebook size

- The codebook vocabulary needs to be much smaller than for doing image search
  - In general, machine-learning techniques need much smaller vectors (for both performance and effectiveness)
  - The visual words can be allowed to be less distinctive, allowing a little more variation between matching features.
  - Typically, the number of visual words might be as small as a few hundred, and up to a few thousand.

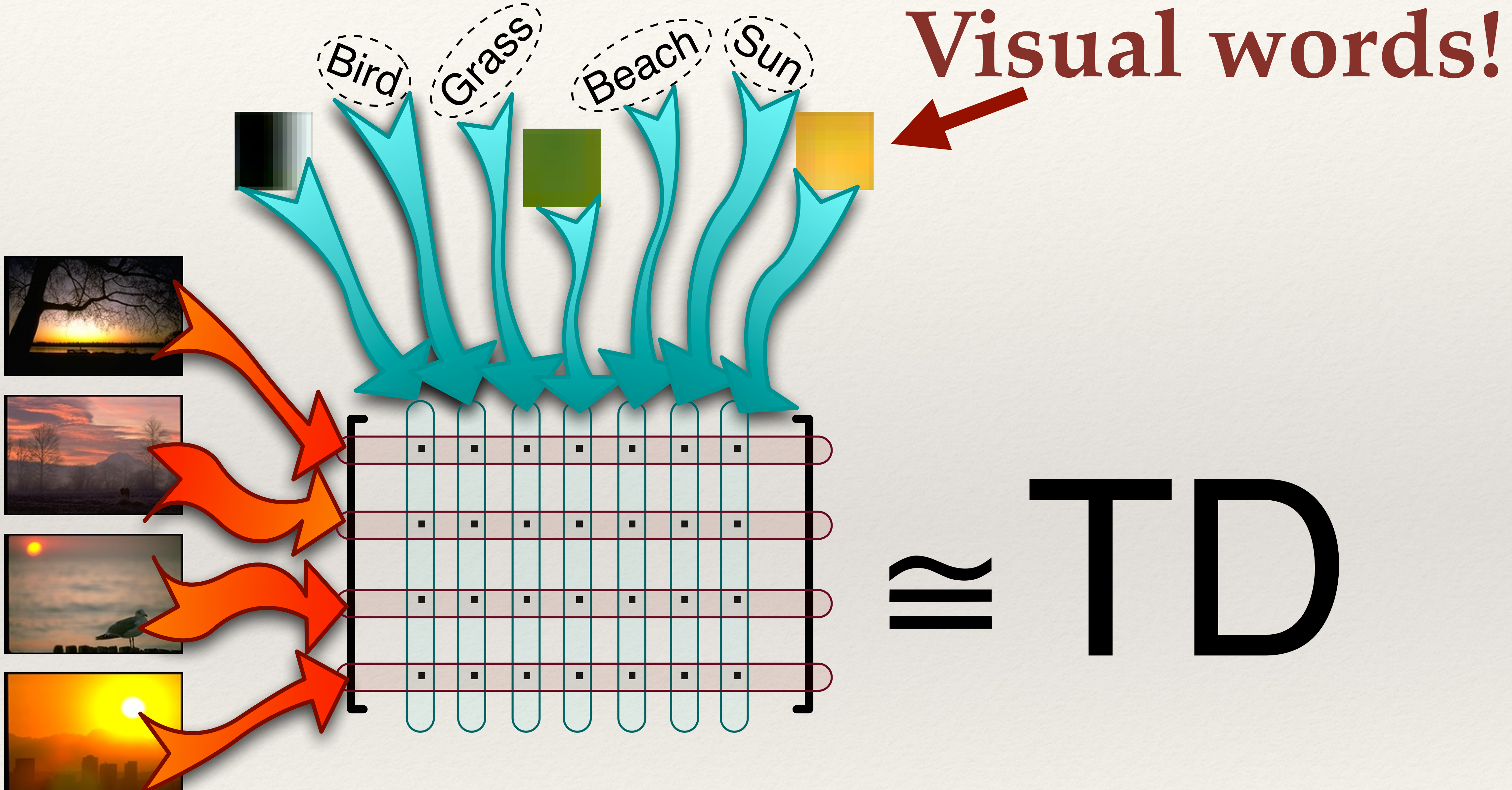
Machine Translation (2002)





Visual words!

Semantic Spaces



Singular Value Decomposition

Probabilistic Latent Factor Models

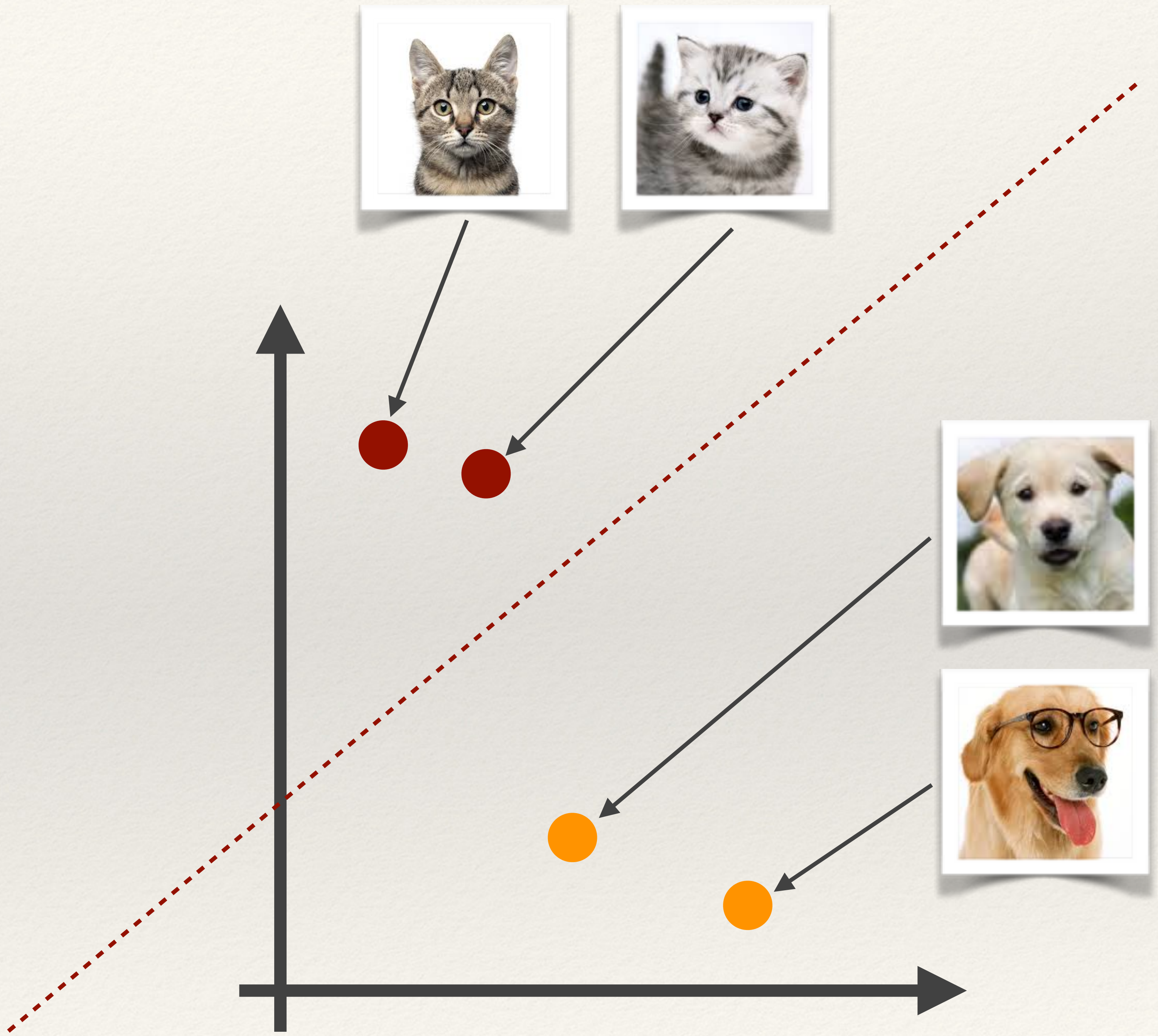
Non-negative Matrix Factorisation

Research focus shifted a little to use of bigger datasets in the mid-late 2000s.

Interest in simpler (but more scalable) classifiers grew

Classifying with BoVW

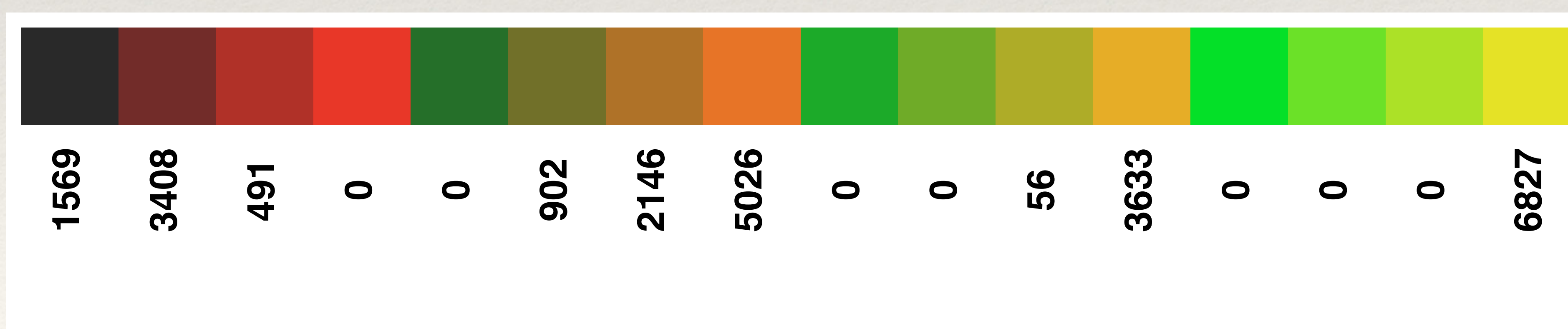
- BoVW histogram representations are incredibly useful for image classification and object detection
  - Commonly used with fast linear classifiers and SVMs



Over time the features used to create BoVW representations have improved

Early global colour visual terms

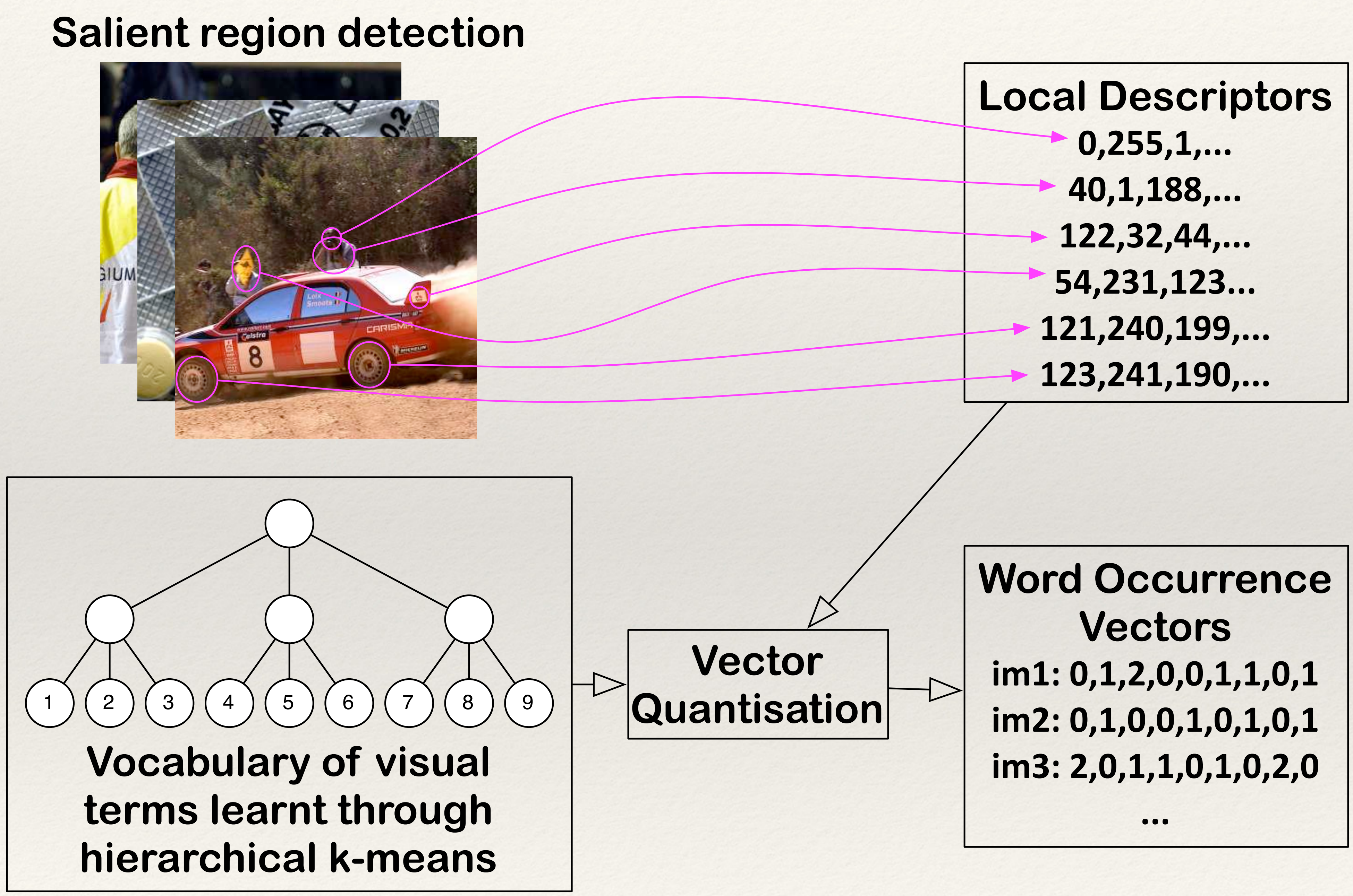
- Consider each pixel as a visual word based on the quantisation of its colour to a discrete set of values.
  - The BoVW Histogram is just a joint colour histogram that we saw earlier



Visual words from regions/segments

[1 2 0 0 6]

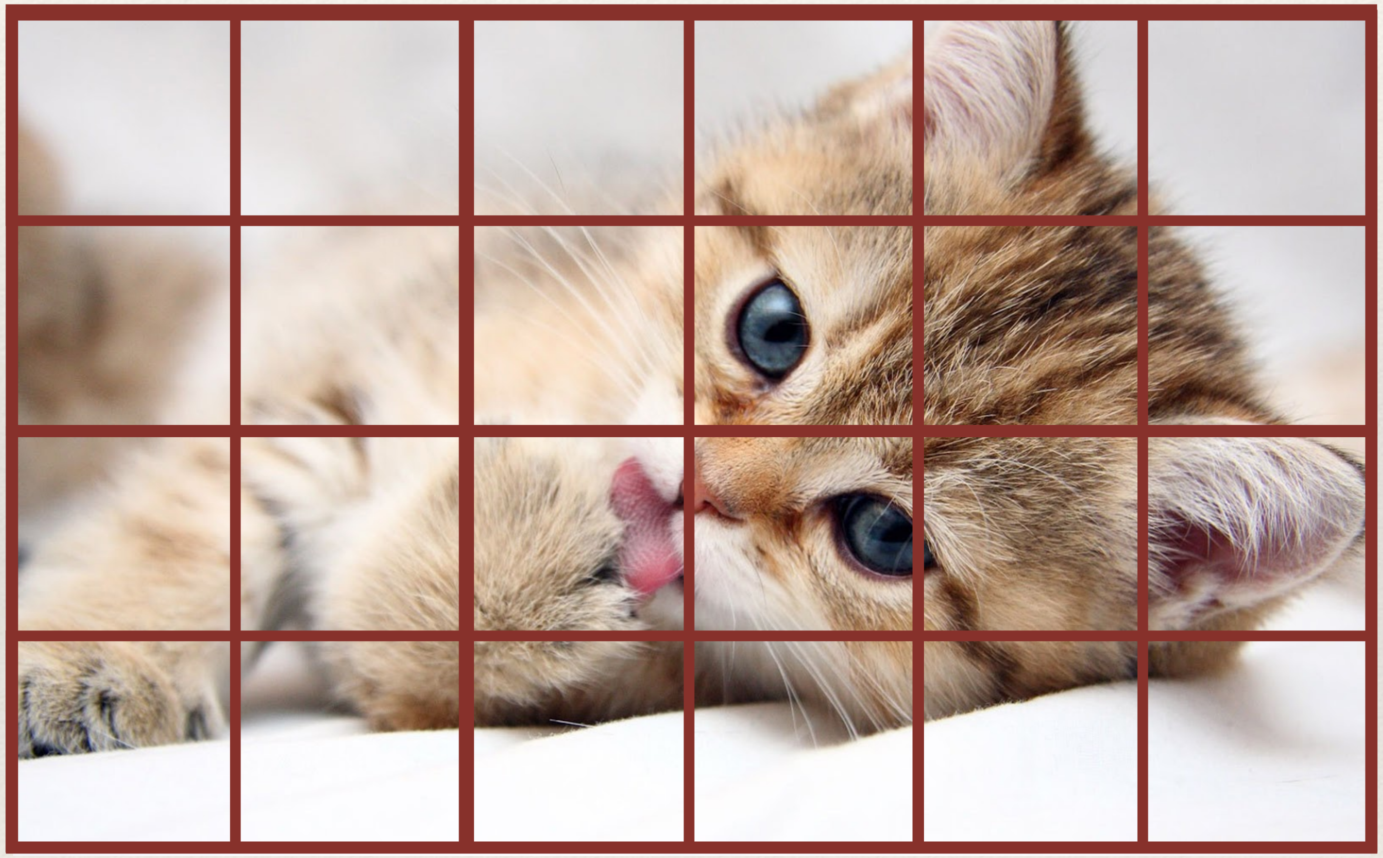
Visual words from interest points



Local features extracted around interest points work okay for classification, but there are more recent strategies that can work better…

densely sampled features

Dense Local Image Patches



Dense SIFT

Rather than extracting your SIFT features at DoG interest points, you could extract them across a dense grid - this gives much more coverage of the entire image.

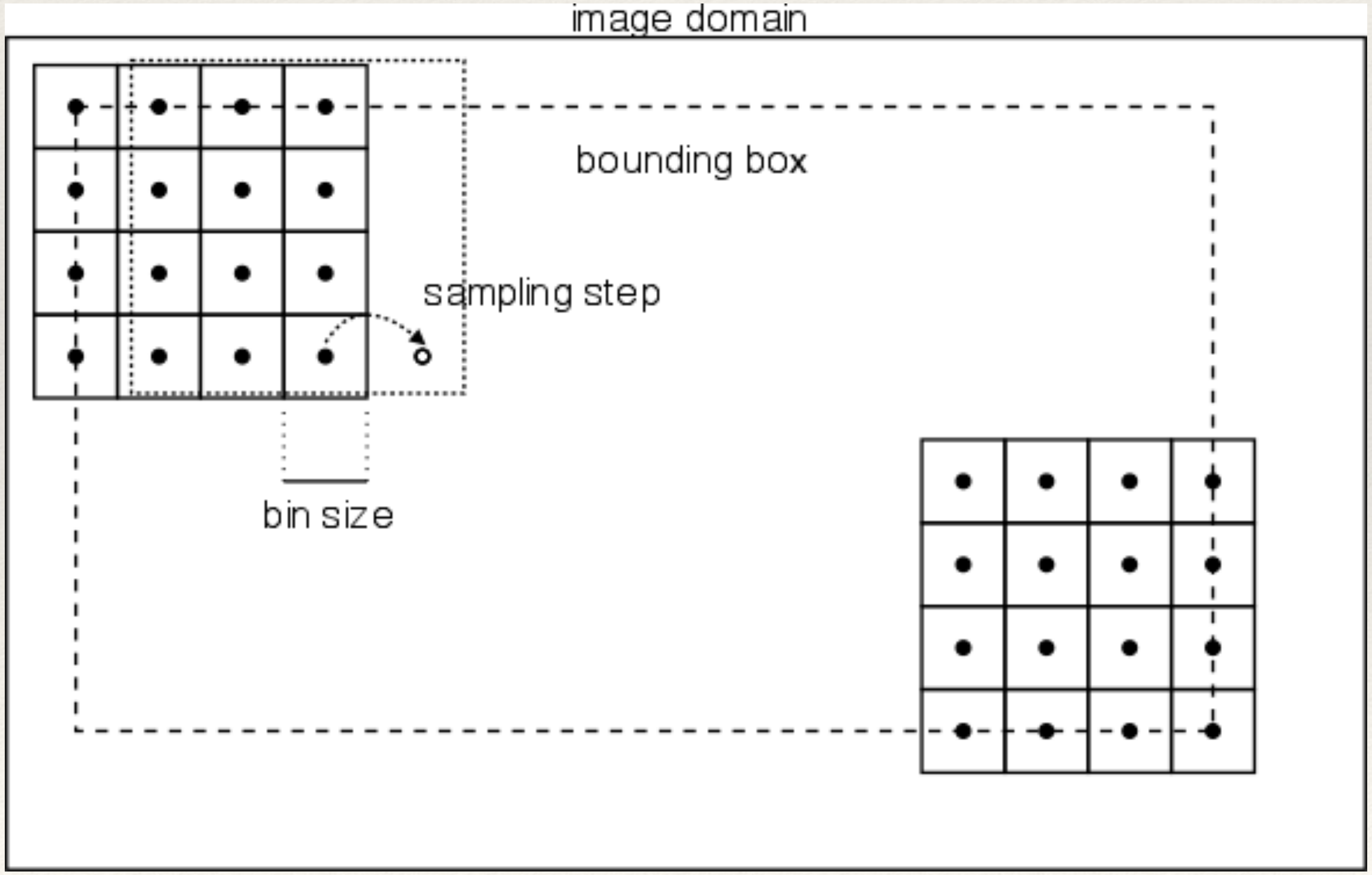
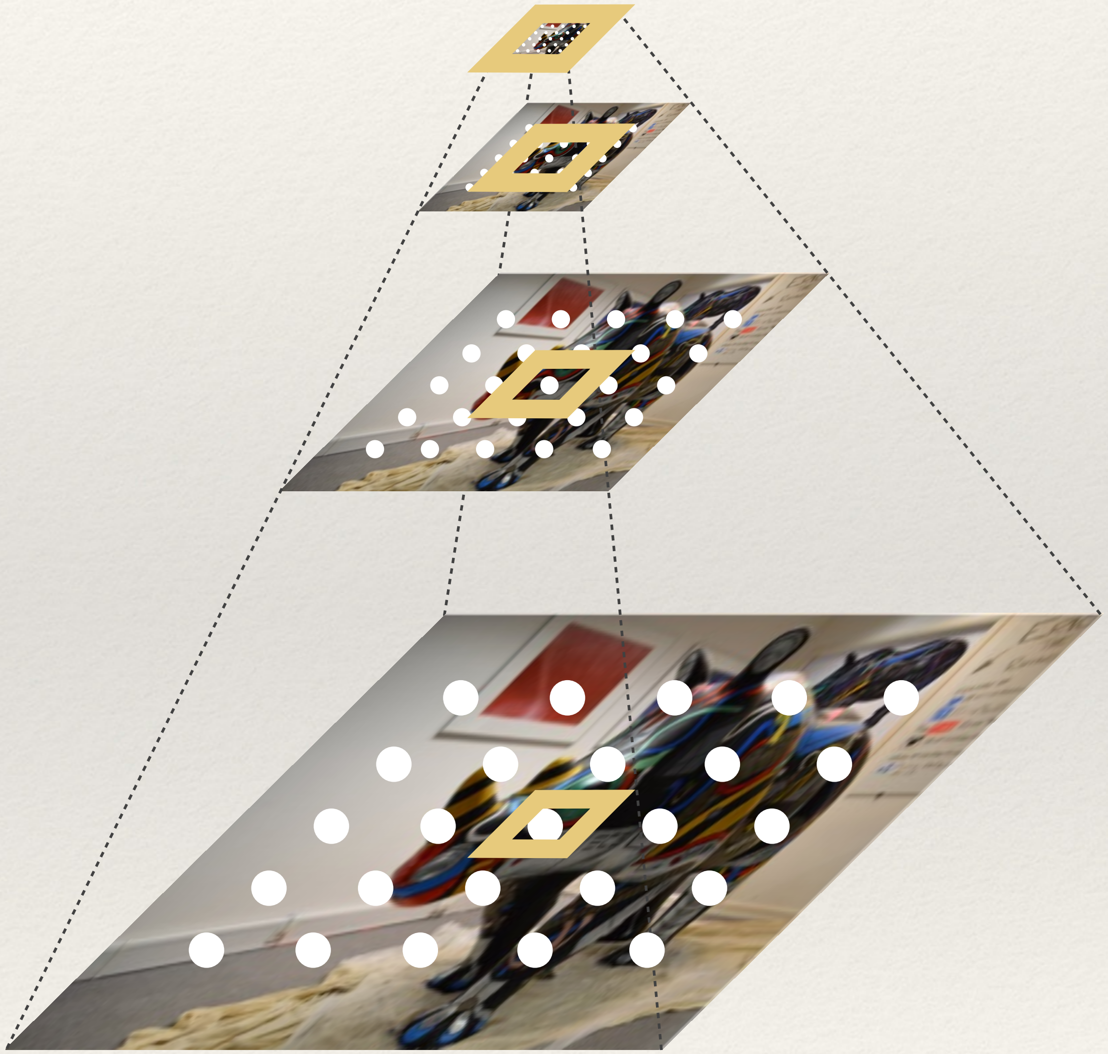


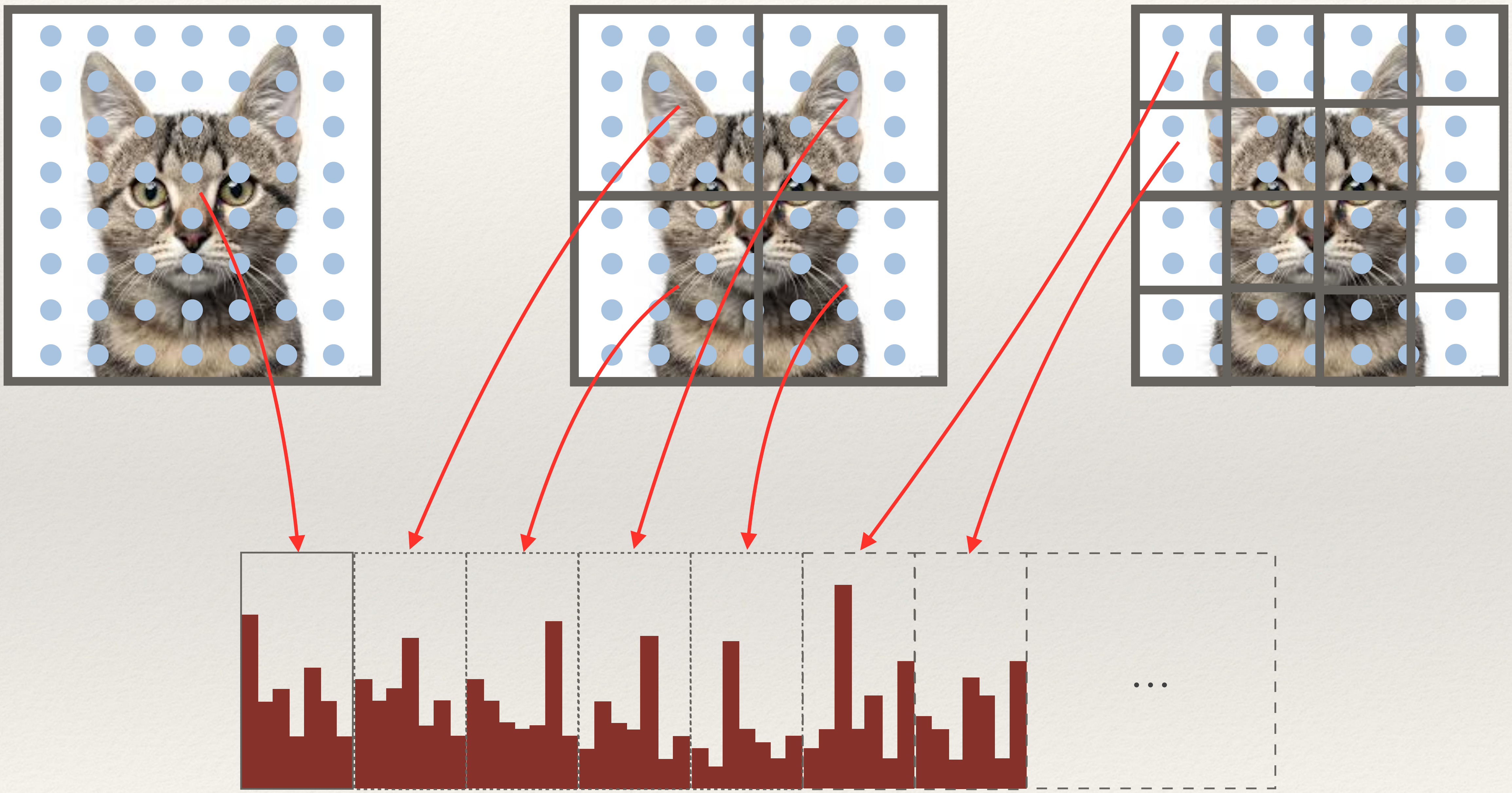
Image from http://www.vlfeat.org/api/dsift.html

Pyramid Dense SIFT

- For even better performance and coverage, you can sample in a Gaussian pyramid
  - Note that the sampling region is a fixed size, so at higher scales you sample more content



Spatial Pyramids



PHOW: Pyramid Histogram of Words = Hist(VQ(Pyramid Dense SIFT)) + Spatial Pyramid

Developing and benchmarking a BoVW scene classifier

Evaluation Dataset

- Common for academic research to use standardised datasets for developing scene classifiers and comparing results
    - Datasets are usually split into labelled "training" and "test" sets.
        - Only the training set can be used to train the classifier
        - Sometimes the test set labels are *withheld* completely to ensure there is no cheating!

Building the BoVW

- Firstly the raw features need to be extracted from the training images
- Then (if necessary) learn a codebook from these features
    - i.e. using k-means on the raw features
        - might be a *uniform random sample* of all the features rather than all of them
- Apply (vector) quantisation to the raw features and count the number of occurrences to build histograms for each image

Training classifiers

- Classifiers can be trained using the histograms.
  - e.g. OvR linear classifiers with a kernel map.
  - You might train on a subset of the training data
    - and use the remaining data to "validate" and optimise parameters.
    - Once you've chosen the optimal parameters you can then re-train using the optimal values.

Classifying the test set

- You're now in a position to apply the classifiers to the test data:
  - Extract the features
  - Quantise the features (using the codebook developed from the training set!)
  - Compute the occurrence histograms
  - Use the classifiers to find the most likely class

Evaluating Performance

- Lots of ways to evaluate performance of classification on the test (and validation) set.
  - Conceptually the simplest summary measure is probably *average precision*
  - this is literally the proportion of number of correct classifications to the total number of predictions

The Final Coursework

Summary

- Object recognition, scene classification and automatic annotation are all important tasks in computer vision.
  - Researchers are striving to narrow the “semantic gap” between what computers can perceive compare to humans.
- The BoVW approach lends itself to high-performance image classification
  - Performance is increased if the local features are sampled densely