Computer Vision 1

THEO GEVERS

MASTER AI

UNIVERSITY OF AMSTERDAM

Lectures/Theory

- 06-02-2018, 17:00-19:00, C0.05, **Introduction** (*Szeliski 1*)
- 13-02-2018, 17:00-19:00, C0.05, **Image Formation** (*Szeliski: 2.1.1 + 2.1.2 + 2.2 + 2.3.2 + 2.3.3*)
- 20-02-2018, 17:00-19:00, C0.05, **Color and Image Processing** (*Szeliski: 3.1 + 3.2 + 3.3*)
- 27-02-2018, 17:00-19:00, C0.05, **Feature Detection, Motion and Classification** (*Szeliski: 4, 8.1.1 + 8.1.3 + 8.2.1 + 8.4; Bengio: 4 + 5.1 + 5.2 + 5.3 + 5.7 + 5.8 + 5.9*)
- 06-03-2018, 17:00-19:00, C0.05, **Object Recognition: BoW and Deep Learning** (*Szeliski: 5.1.1 + 5.1.4 + 5.1.5 + 5.2 + 5.3 + 5.4, 6.1 + 6.3, 14.1 + 14.2.1 + 14.3 + 14.4.1; Bengio: 7.2 + 7.4 + 9.1 + 9.2 + 9.3*)
- 13-03-2018, 17:00-19:00, C0.05, **ConvNets, Stereo and 3D Reconstruction** (*Szeliski: 11.1 + 11.2 + 11.3 + 11.4, 12.1 + 12.2: Bengio: 12.1 + 12.2*)
- 20-03-2018, 17:00-19:00, C0.05, **Applications** (*Szeliski: 12.6.2 + 12.6.3 + 12.2.4)*
- 26-03-2018, Monday, 9:00-12:00, **Written Exam**

Today's class

Segmentation (short)

Object Recognition

Bag-of-Words

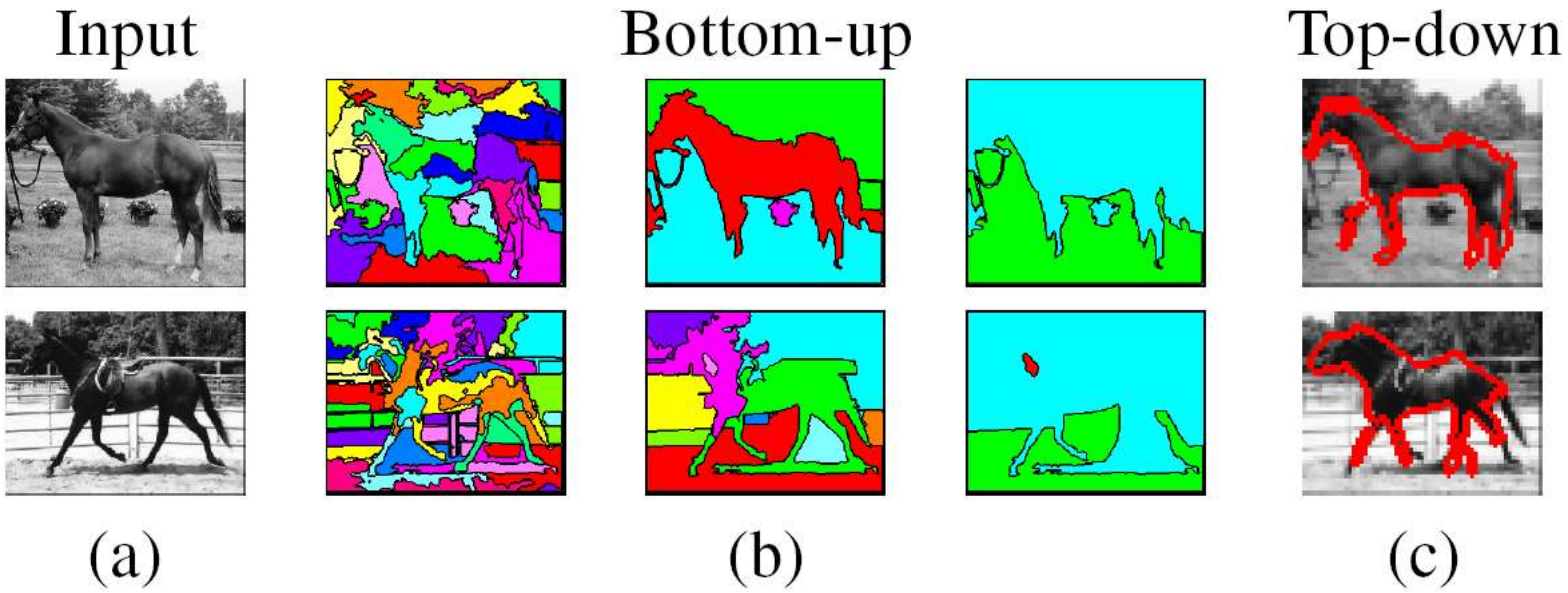
Deep Learning

Image Segmentation

Goal: Break up the image into meaningful or perceptually similar regions

Major Processes for Segmentation

- Bottom-up: group regions with similar features
- Top-down: split regions that likely belong to a different object



Slide: Levin and Weiss 2006

How Do We Segment?

- K-means
  - Iteratively re-assign points to the nearest cluster center
- Split&Merge
  - Split the image and merge regions afterwards
- Mean-shift clustering
  - Estimate modes of pdf
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights

K-means (Recap)

1. Initialize cluster centers: $\mathbf{c}^0$; t=0

2. Assign each point to the closest center

$$\boldsymbol{\delta}^t = \underset{\boldsymbol{\delta}}{\operatorname{argmin}} \frac{1}{N} \sum_j^N \sum_i^K \delta_{ij} \left(\mathbf{c}_i^{t-1} - \mathbf{x}_j \right)^2$$

3. Update cluster centers as the mean of the points

$$\mathbf{c}^t = \underset{\mathbf{c}}{\operatorname{argmin}} \frac{1}{N} \sum_j^N \sum_i^K \delta_{ij}^t \left(\mathbf{c}_i - \mathbf{x}_j \right)^2$$

4. Repeat 2-3 until no points are re-assigned (t=t+1)

Slide: Derek Hoiem

K-means: Design Choices

- Initialization
  - Randomly select K points as initial cluster center
- Distance measures
  - Traditionally Euclidean, could be others
- Optimization
  - Will converge to a *local minimum*
  - May want to perform multiple restarts

K-means Clustering: Intensity or Color

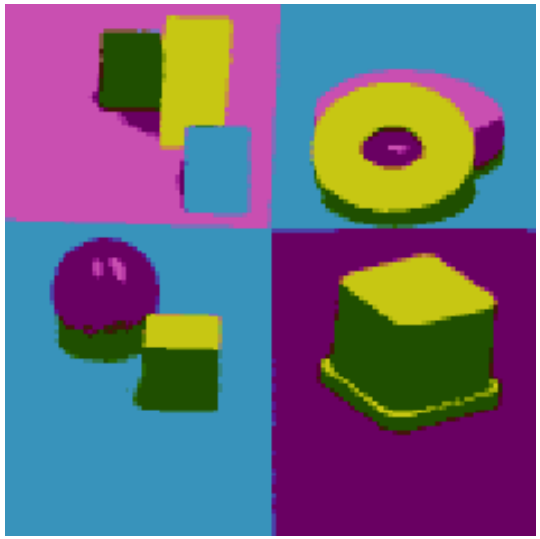
Image

Clusters on intensity

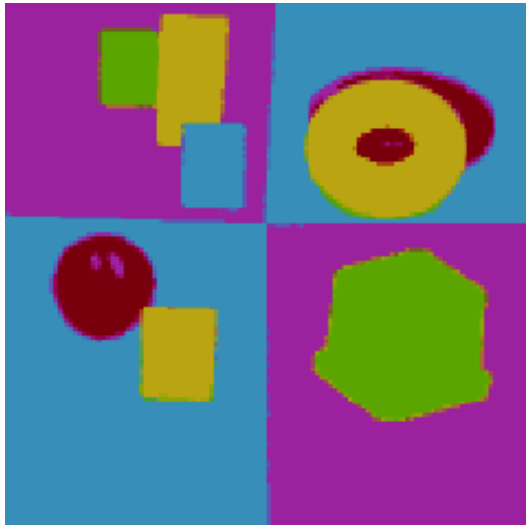
Clusters on color

K-means

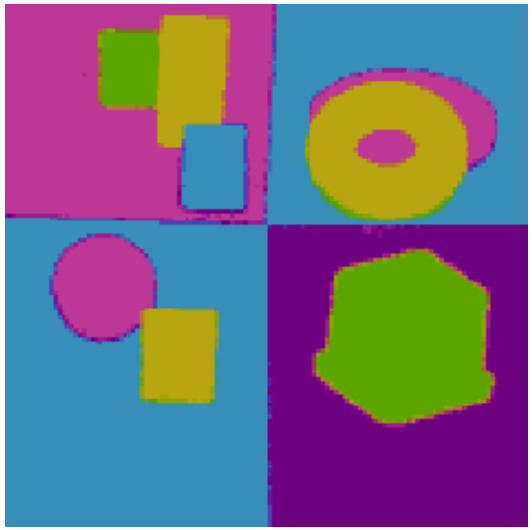
Original image



Mean&var in RGB

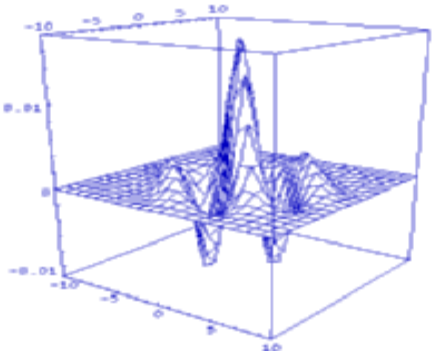


Mean&var in rgb



Mean&var in H

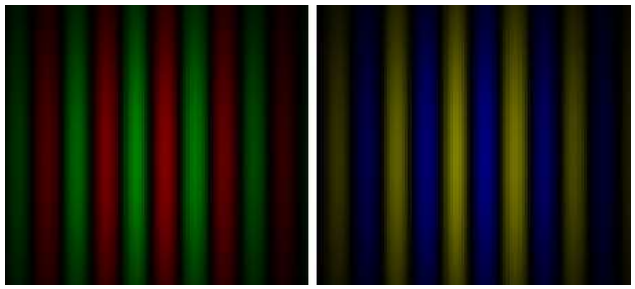
Texture

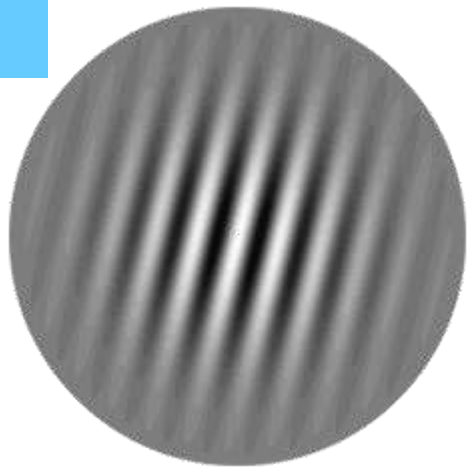


The 2D Gabor function is:

$$h(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\delta^2}} e^{2\pi j(ux+vy)}$$

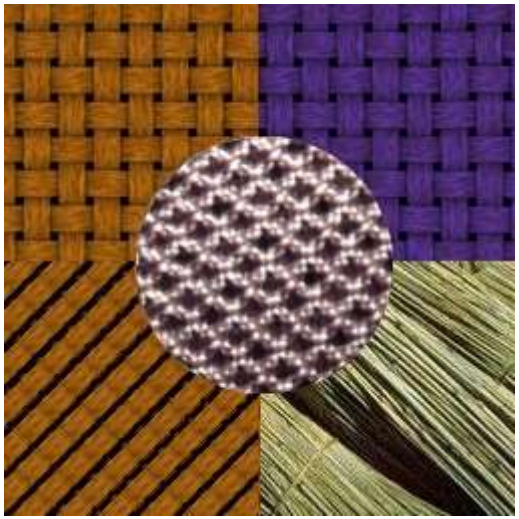
Tuning parameters: *u, v*
+ usual invariants by combination



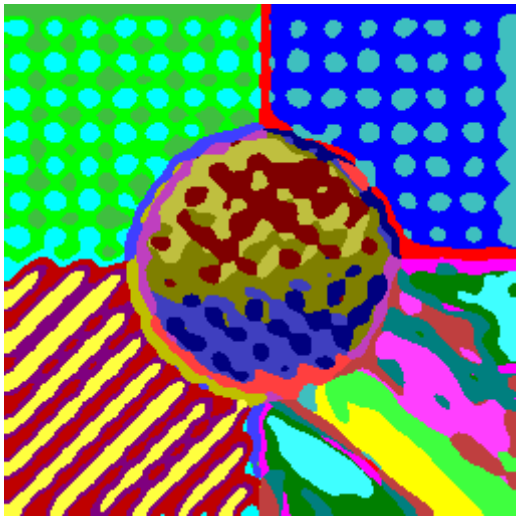


Minh SP 2005

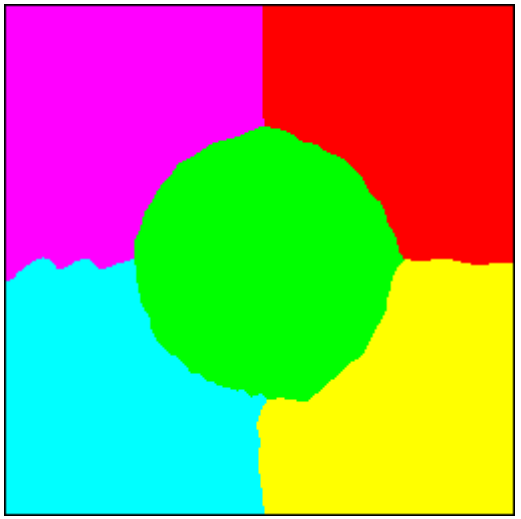
Gabor and K-means Clustering



Original image



K-means clustering



Segmentation

Minh, ECCV, 2002

Gabor and K-means Clustering

How Do We Segment?

- K-means
  - Iteratively re-assign points to the nearest cluster center
- ***Split&Merge***
  - ***Iteratively split and merge regions***
- Mean-shift clustering
  - Estimate modes of pdf
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights

Split-and-Merge

Split regions until patch is homogeneous wrt query image

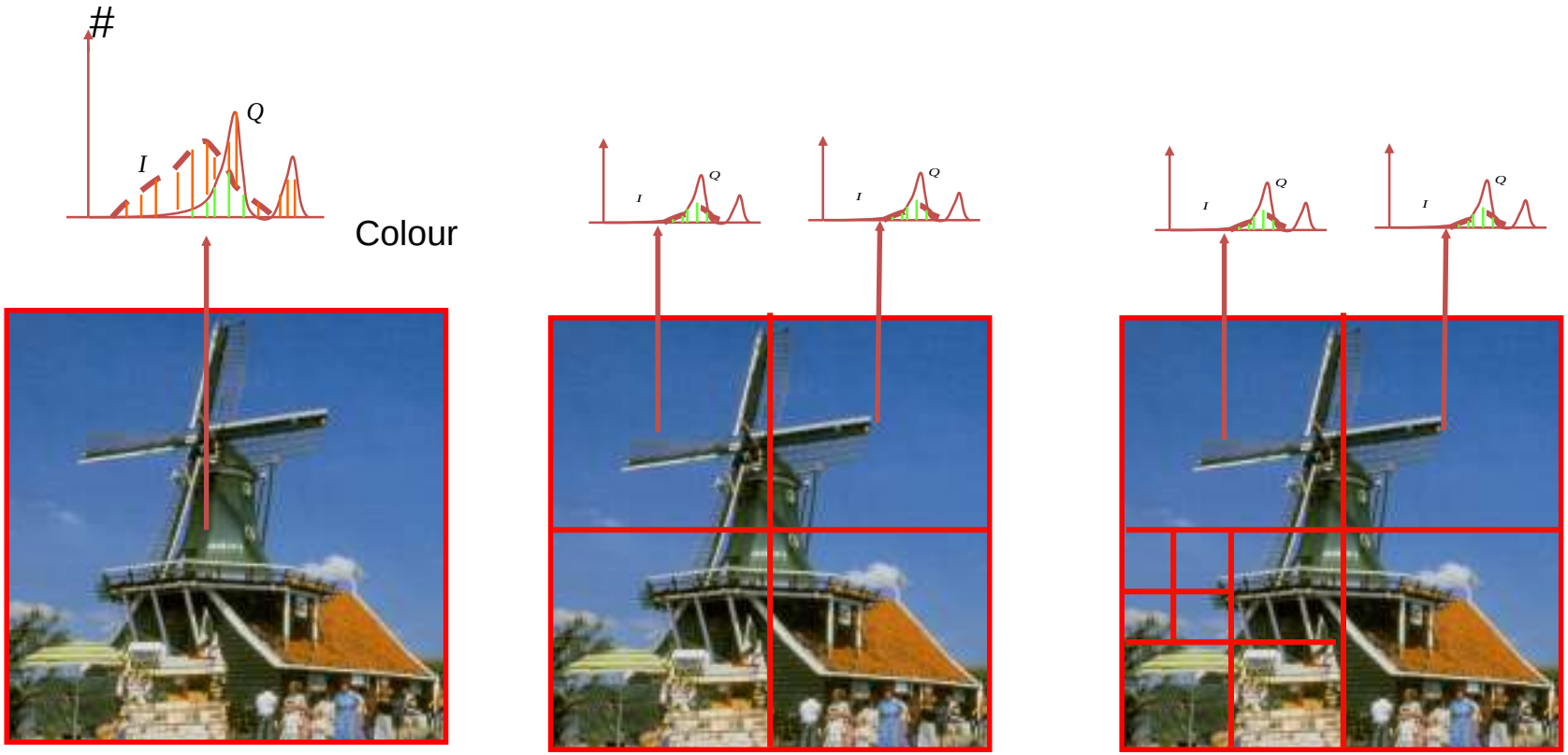
Split-and-Merge

... and merge patches which are alike.

It works because of spatial coherence.

Split-and-Merge

Split regions until patch is homogeneous wrt query image

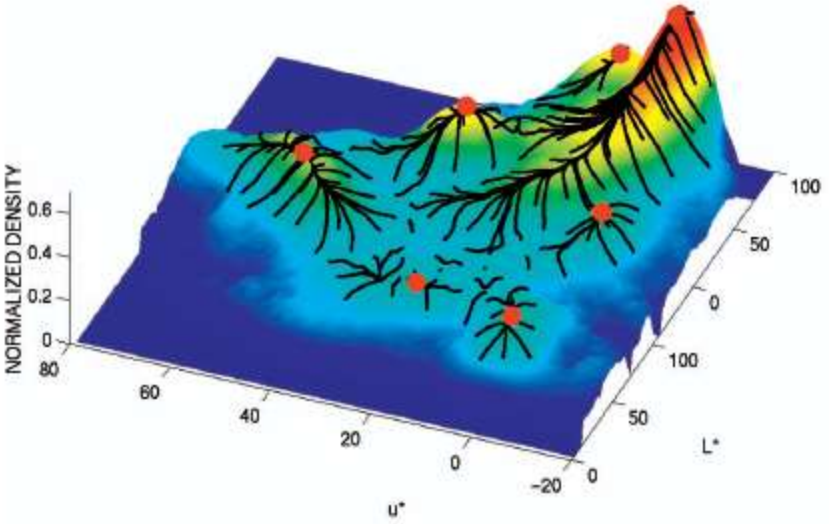


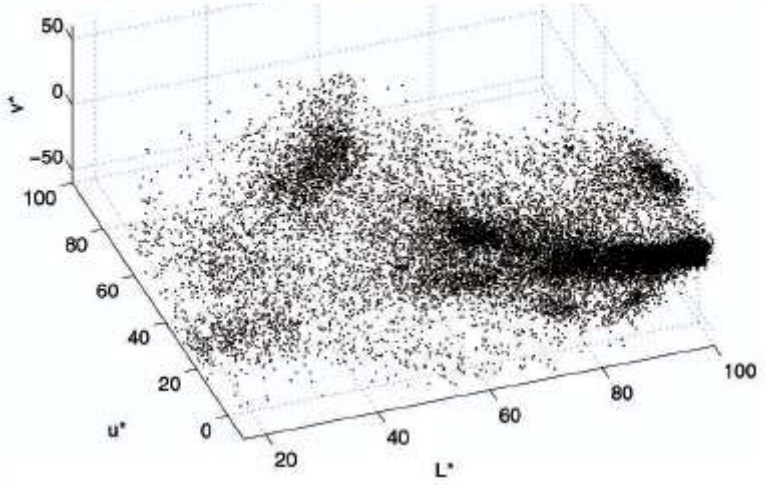
How Do We Segment?

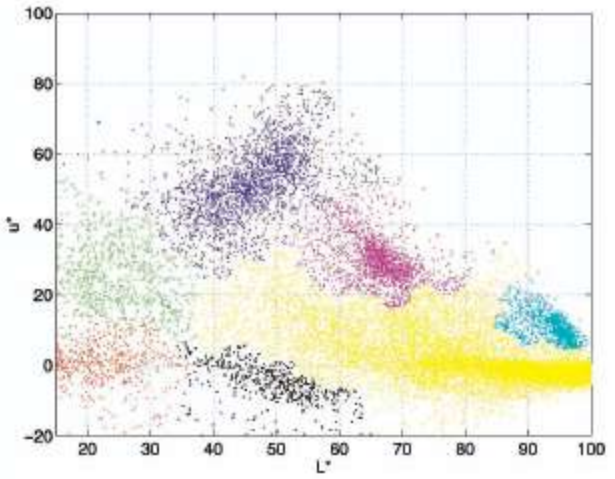
- K-means
  - Iteratively re-assign points to the nearest cluster center
- Split&Merge
  - Iteratively split and merge regions
- Mean-shift clustering
  - Estimate modes of pdf
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights

Mean Shift Algorithm

- Try to find *modes* of this non-parametric density







Mean Shift Segmentation Results

http://www.caip.rutgers.edu/~comanici/MSPAMI/msPamiResults.html

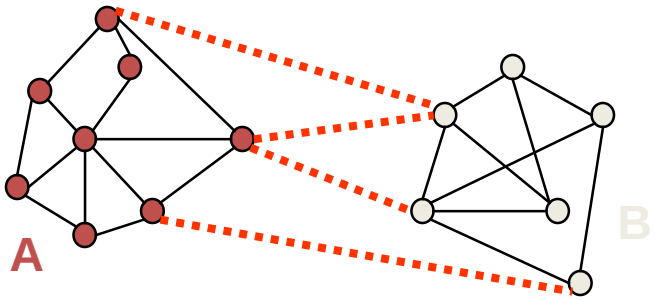
Mean Shift Pros and Cons

- Pros
  - Good general-practice segmentation
  - Flexible in number and shape of regions
  - Robust to outliers
- Cons
  - Have to choose kernel size in advance
  - Not suitable for high-dimensional features
- When to use it
  - Oversegmentatoin
  - Multiple segmentations
  - Tracking, clustering, filtering applications

How Do We Segment?

- K-means
  - Iteratively re-assign points to the nearest cluster center
- Split&Merge
  - Iteratively split and merge regions
- Mean-shift clustering
  - Estimate modes of pdf
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights

Cuts in a Graph



Normalized Cut

- a cut penalizes large segments
- fix by normalizing for size of segments

$$Ncut(A,B) = \frac{cut(A,B)}{volume(A)} + \frac{cut(A,B)}{volume(B)}$$

- volume(A) = sum of costs of all edges that touch A

Source: Seitz

Normalized Cuts for Segmentation

Today's class

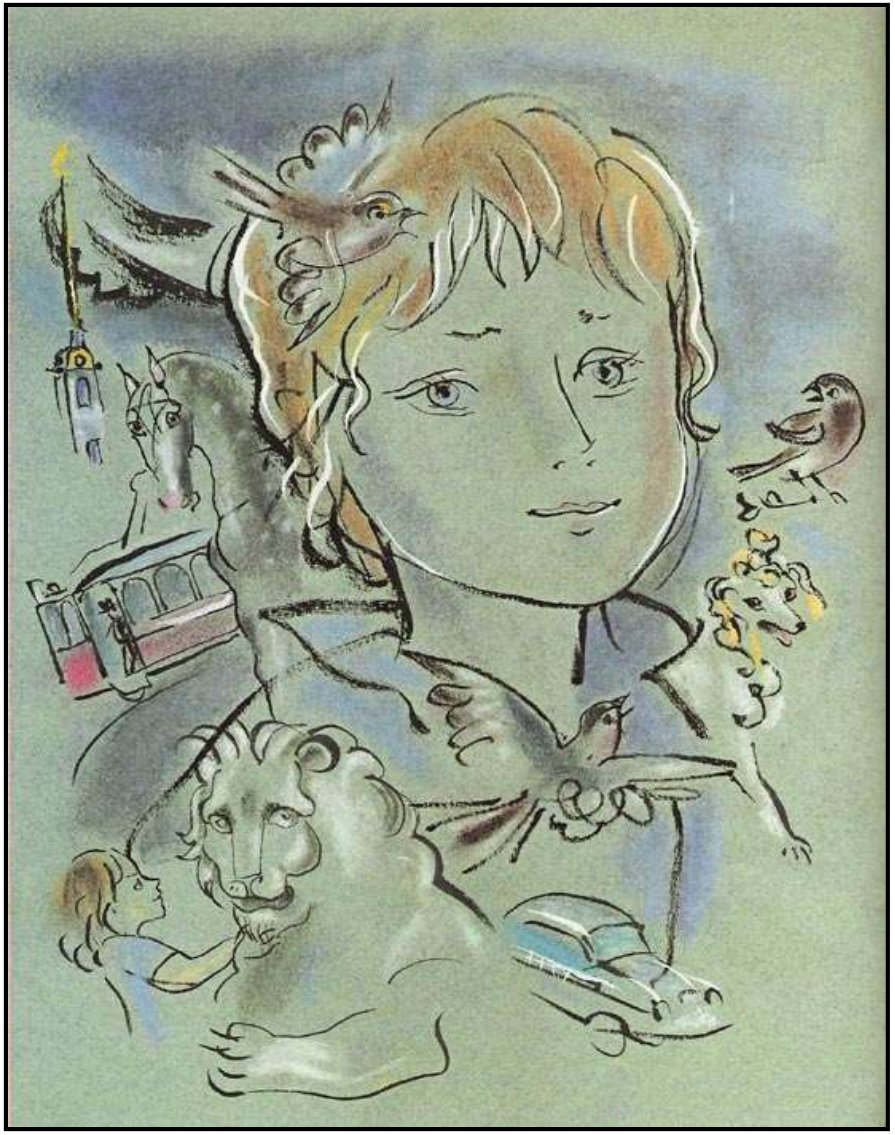
Segmentation

Object Recognition

Bag-of-Words

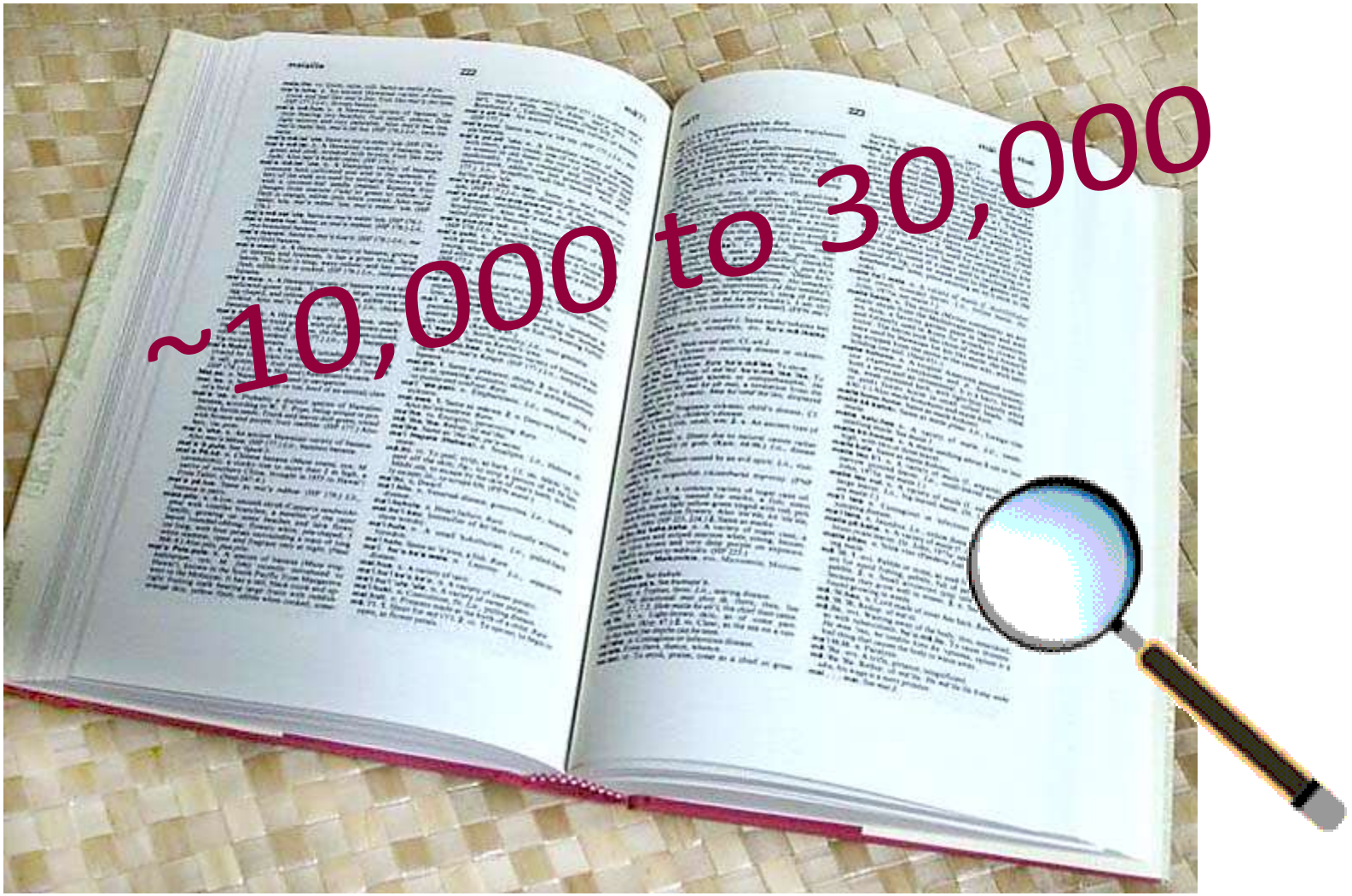
Deep Learning

Recognition: Overview and History



Slides from Lana Lazebnik, Fei-Fei Li, Rob Fergus, Antonio Torralba, and Jean Ponce

How Many Visual Object Categories?



Slide: Biederman

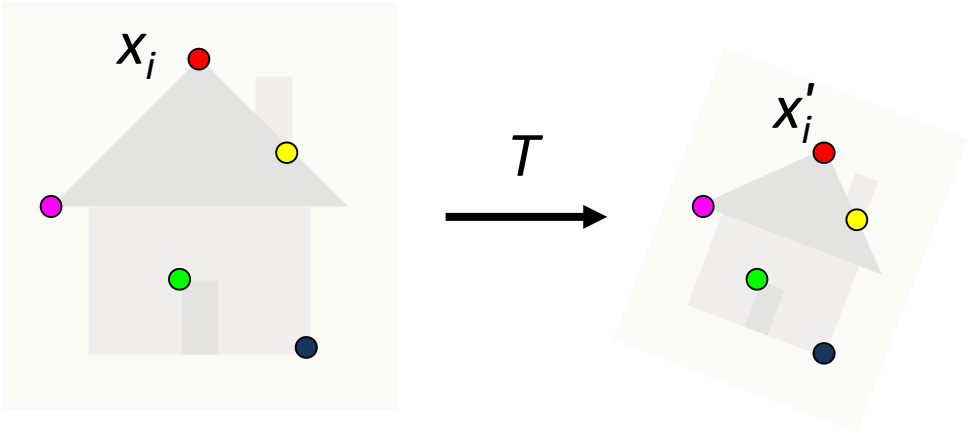

~10,000 to 30,000

History of Ideas in Object Recognition

- 1960s – early 1990s: the geometric era

Geometric Alignment

- Alignment: fitting a model to a transformation between pairs of features (*matches*) in two images



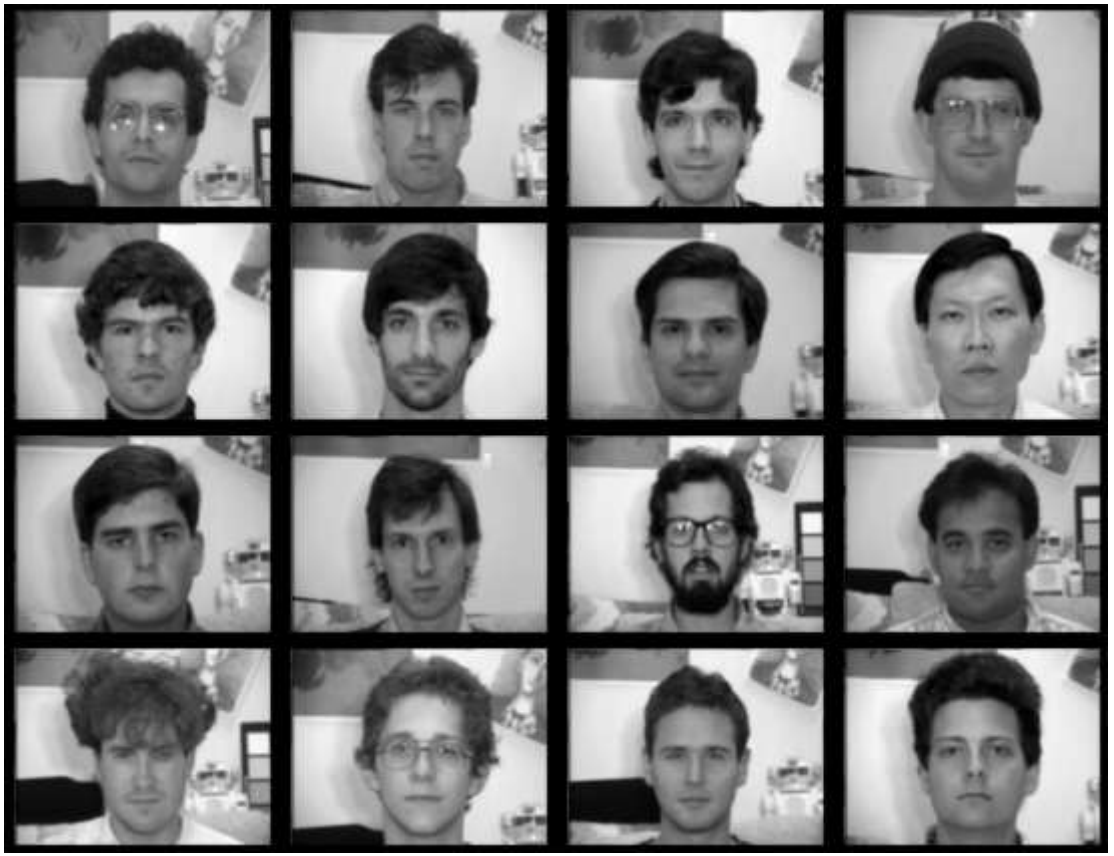
Find transformation T that minimizes

$$\sum_i \text{residual}(T(x_i), x_i')$$

History of Ideas in Object Recognition

- 1960s – early 1990s: the geometric era
- 1990s: appearance-based models

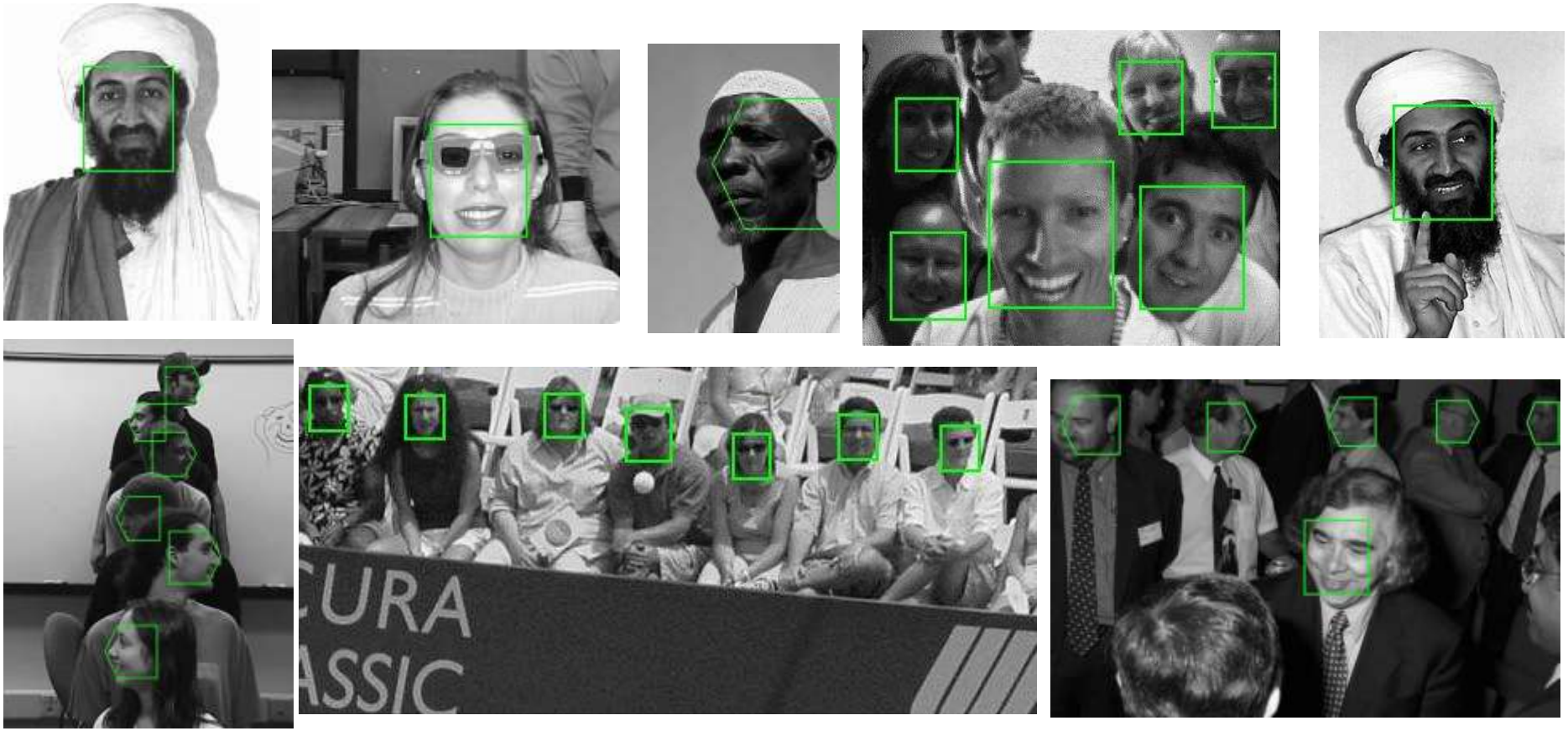
Eigenfaces (Turk & Pentland, 1991)



Experimental	Correct/Unknown Recognition Percentage		
Condition	Lighting	Orientation	Scale
Forced classification	96/0	85/0	64/0
Forced 100% accuracy	100/19	100/39	100/60
Forced 20% unknown rate	100/20	94/20	74/20

Face Detection

Schneiderman & Kanade (CMU), 2000…



Results on various images submitted to the CMU on-line face detector

History of Ideas in Object Recognition

- 1960s – early 1990s: the geometric era
- 1990s: appearance-based models
- 1990s – present: sliding window approaches

Sliding Window Approaches

Sliding Window Approaches

- Turk and Pentland, 1991
- Belhumeur, Hespanha, & Kriegman, 1997
- Schneiderman & Kanade 2004
- Viola and Jones, 2000

- Schneiderman & Kanade, 2004
- Argawal and Roth, 2002
- Poggio et al. 1993

History of Ideas in Object Recognition

- 1960s – early 1990s: the geometric era
- 1990s: appearance-based models
- Mid-1990s: sliding window approaches
- Late 1990s: local features

Local Features for Object Recognition

D. Lowe (1999, 2004)

History of Ideas in Object Recognition

- 1960s – early 1990s: the geometric era
- 1990s: appearance-based models
- Mid-1990s: sliding window approaches
- Late 1990s: local features
- Early 2000s: parts-and-shape models

Constellation Models

Weber, Welling & Perona (2000), Fergus, Perona & Zisserman (2003)

History of Ideas in Object Recognition

- 1960s – early 1990s: the geometric era
- 1990s: appearance-based models
- Mid-1990s: sliding window approaches
- Late 1990s: local features
- Early 2000s: parts-and-shape models
- Mid-2000s: bags-of-features
- Present trends: Deep Learning

Today's class

Segmentation

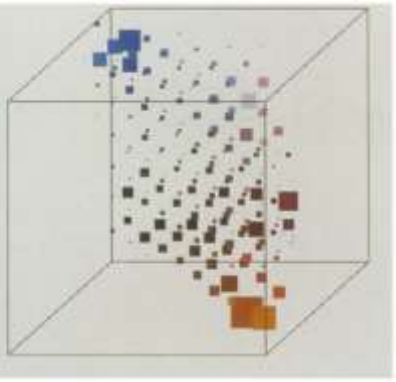
Object Recognition

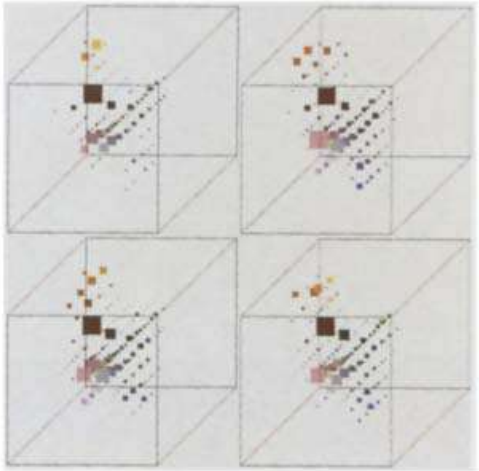
Bag-of-Words

Deep Learning

Object Recognition using Color Histograms

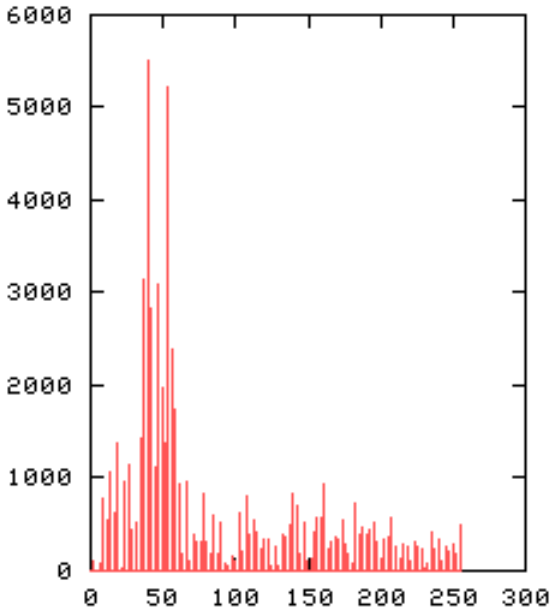
Color Histograms

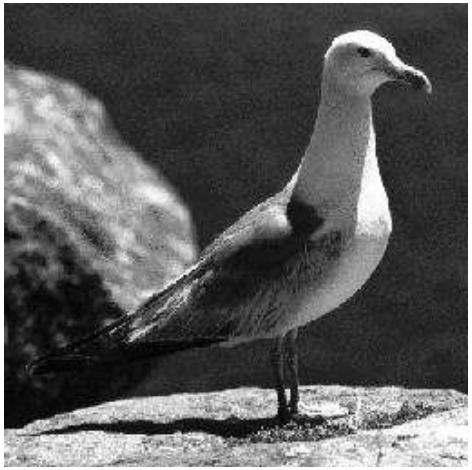




Swain and Ballard, Color Indexing, IJCV 1991.

Image Representations: Histograms





Global histogram

- Represent distribution of features
  - Color, texture, depth, …

Images from Dave Kauchak

Computing Histogram Distance

$$\text{histint}(h_i, h_j) = 1 - \sum_{m=1}^{K} \min\left(h_i(m), h_j(m)\right)$$

Histogram intersection (assuming normalized histograms)

$$\chi^2(h_i, h_j) = \frac{1}{2} \sum_{m=1}^{K} \frac{[h_i(m) - h_j(m)]^2}{h_i(m) + h_j(m)}$$

Chi-squared Histogram matching distance

Cars found by color histogram matching using chi-squared

Histograms: Implementation Issues

- Quantization
  - Grids: fast but applicable only with few dimensions
  - Clustering: slower but can quantize data in higher dimensions

Few Bins
Need less data
Coarser representation

Many Bins
Need more data
Finer representation

- Matching
  - Histogram intersection or Euclidean may be faster
  - Chi-squared often works better

Object Recognition using K-Nearest Neighbors

1. Collect a dataset of images and labels
2. Use Machine Learning to train an image classifier
3. Evaluate the classifier on a withheld set of test images

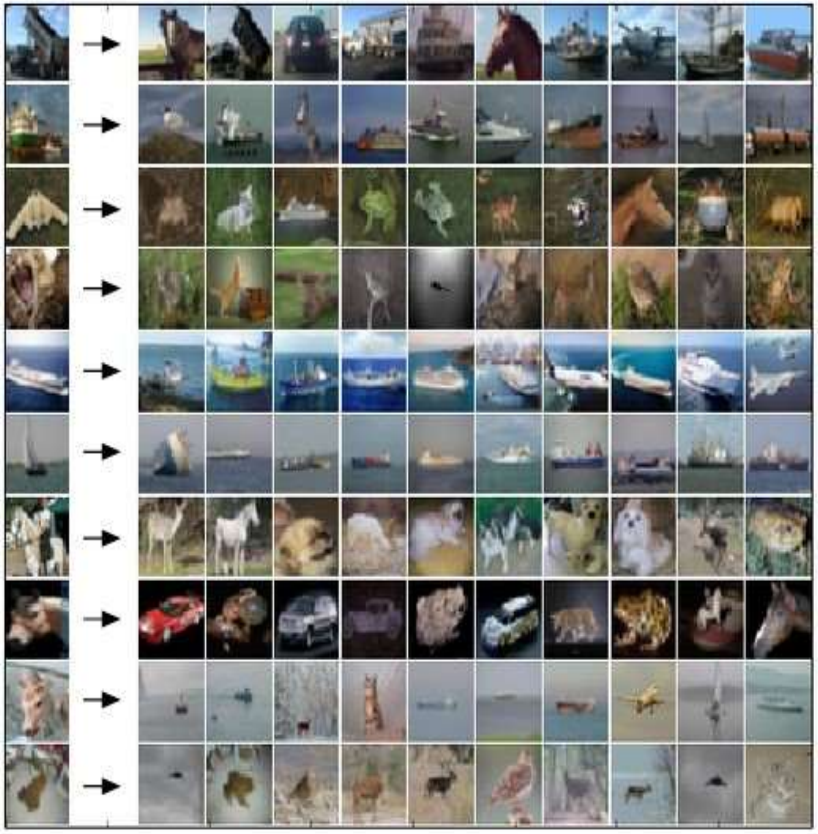
Example training set

Dataset: CIFAR-10

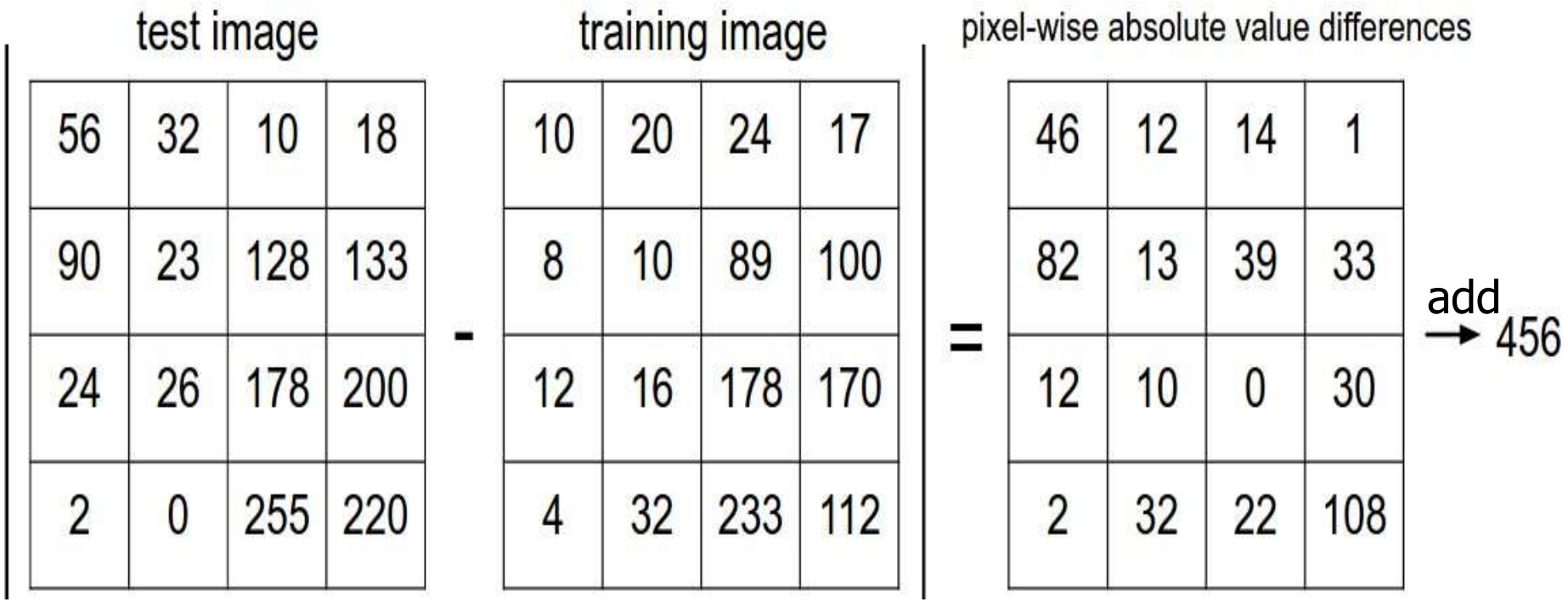
10 labels
50,000 training images, each image is tiny: 32x32
10,000 test images.

10 labels
50,000 training images
10,000 test images.

For every test image (first column), examples of nearest neighbors in rows



L1 distance: $$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$



The choice of distance is a **hyperparameter**

L1 (Manhattan) distance

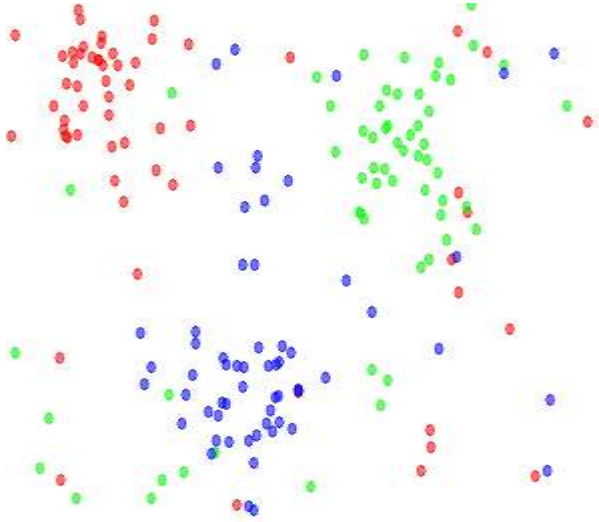
$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$

L2 (Euclidean) distance

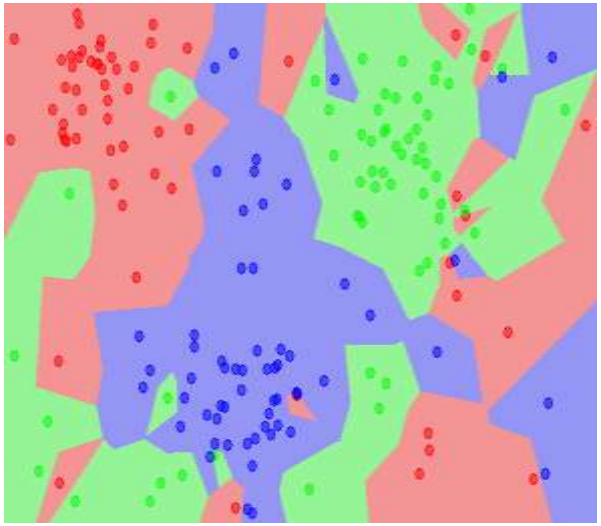
$$d_2(I_1, I_2) = \sqrt{\sum_p \left(I_1^p - I_2^p\right)^2}$$

The choice of distance is a **hyperparameter**

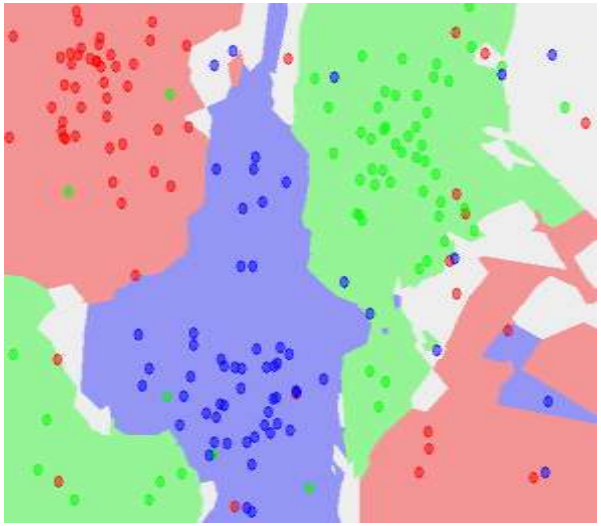
the data



NN classifier



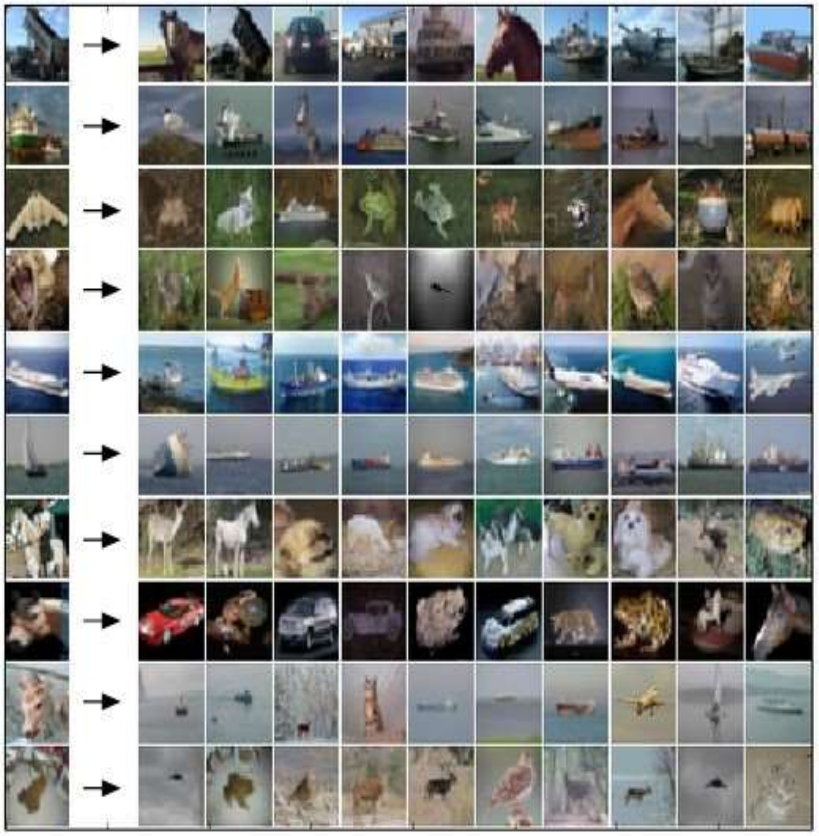
5-NN classifier



http://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

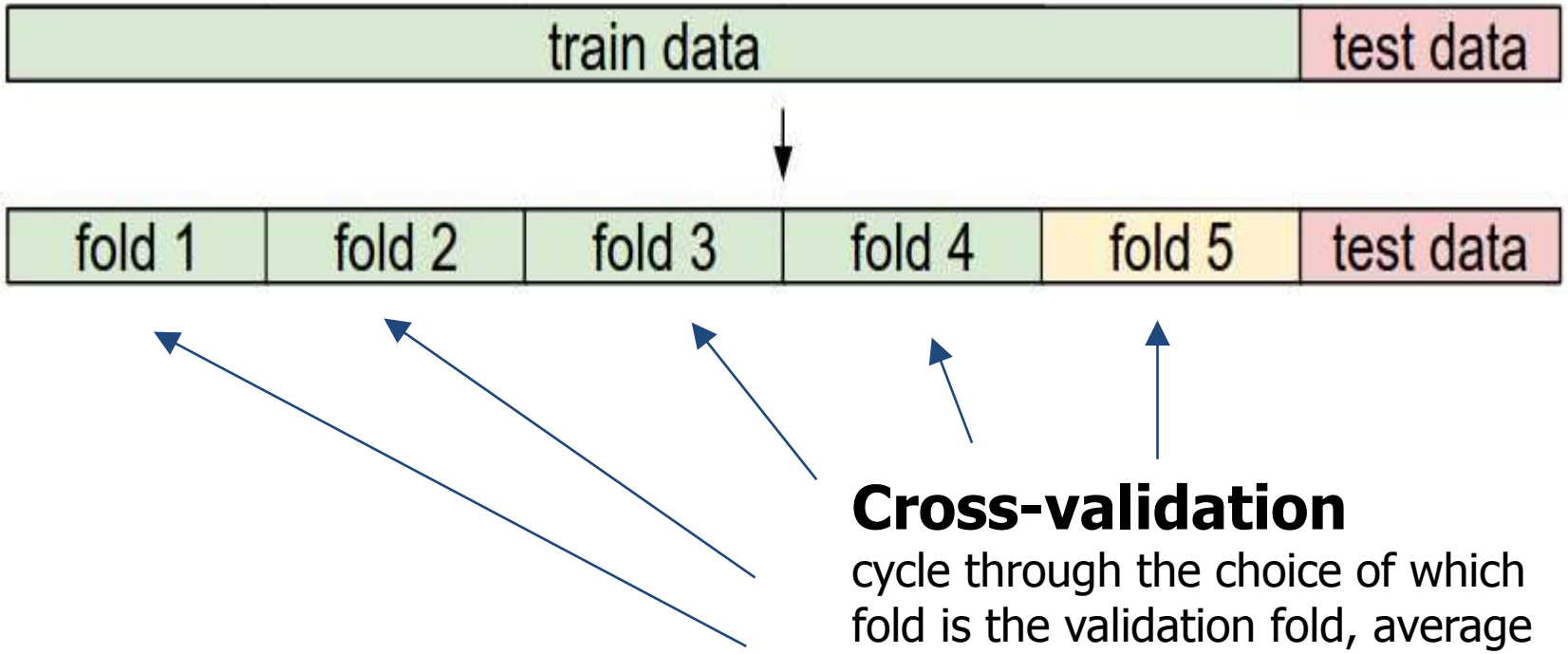
10 labels
50,000 training images
10,000 test images.

For every test image (first column), examples of nearest neighbors in rows



What is the best **distance** to use?
What is the best value of **k** to use?

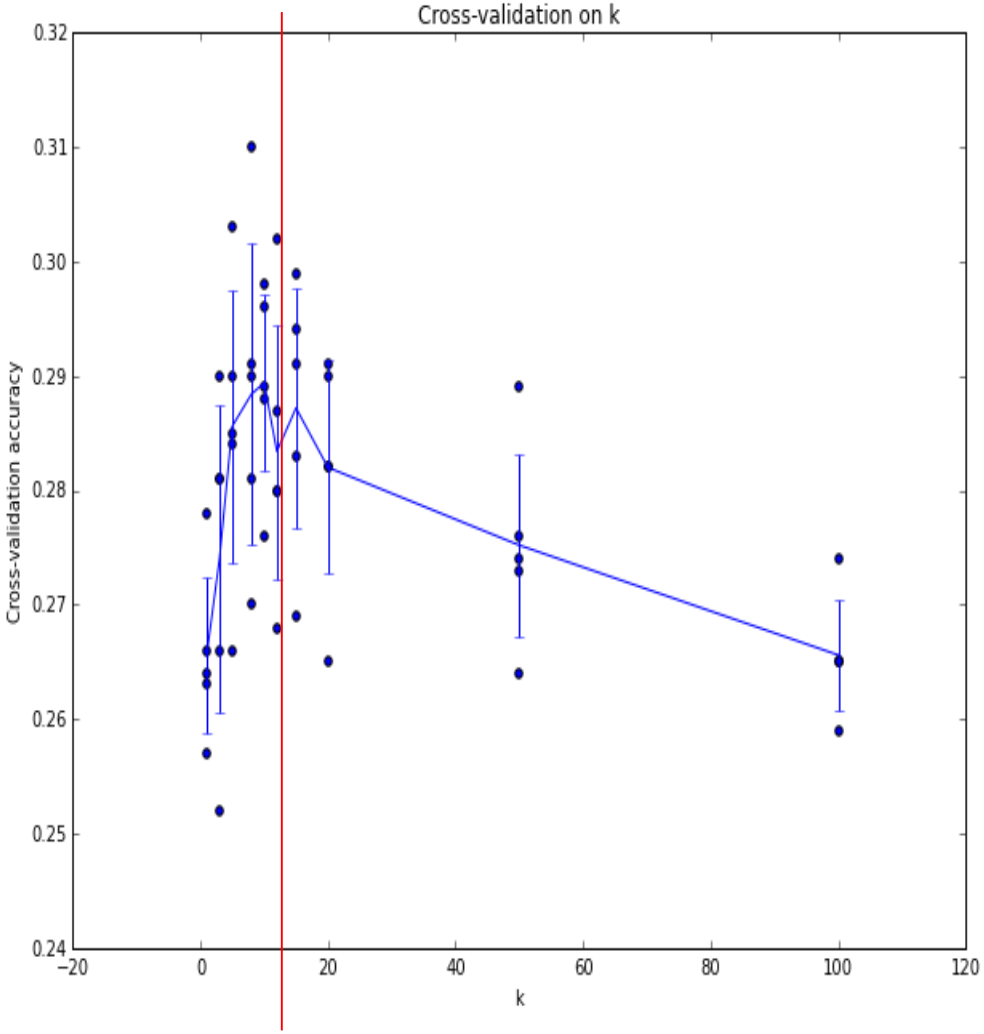
i.e. how do we set the **hyperparameters**?



Cross-validation

cycle through the choice of which fold is the validation fold, average results.

Andrej Karpathy, Fei-Fei Li, Justin Johnson



Example of
5-fold cross-validation
for the value of **k.**

Each point: single
outcome.

The line goes
through the mean, bars
indicated standard
deviation

(Seems that k ~= 7 works best
for this data)

- **Image Classification:** given a **Training Set** of labeled images, predict labels on **Test Set**. Common to report the **Accuracy** of predictions (fraction of correct predictions)
- We introduced the **k-Nearest Neighbor Classifier**, which predicts labels based on nearest images in the training set
- We saw that the choice of distance and the value of k are **hyperparameters** that are tuned using a **validation set**, or through **cross-validation** if the size of the data is small.
- Once the best set of hyperparameters is chosen, the classifier is evaluated once on the test set, and reported as the performance of kNN on that data.

Today's class

Segmentation

Object Recognition

Bag-of-Words

Deep Learning

Object Recognition based on BoW

Bag-of-features Models

Object → **Bag of 'words'**

Image Categorization

Training

Training Images

Training Labels

Image Features

Classifier Training

Trained Classifier

Testing

Test Image

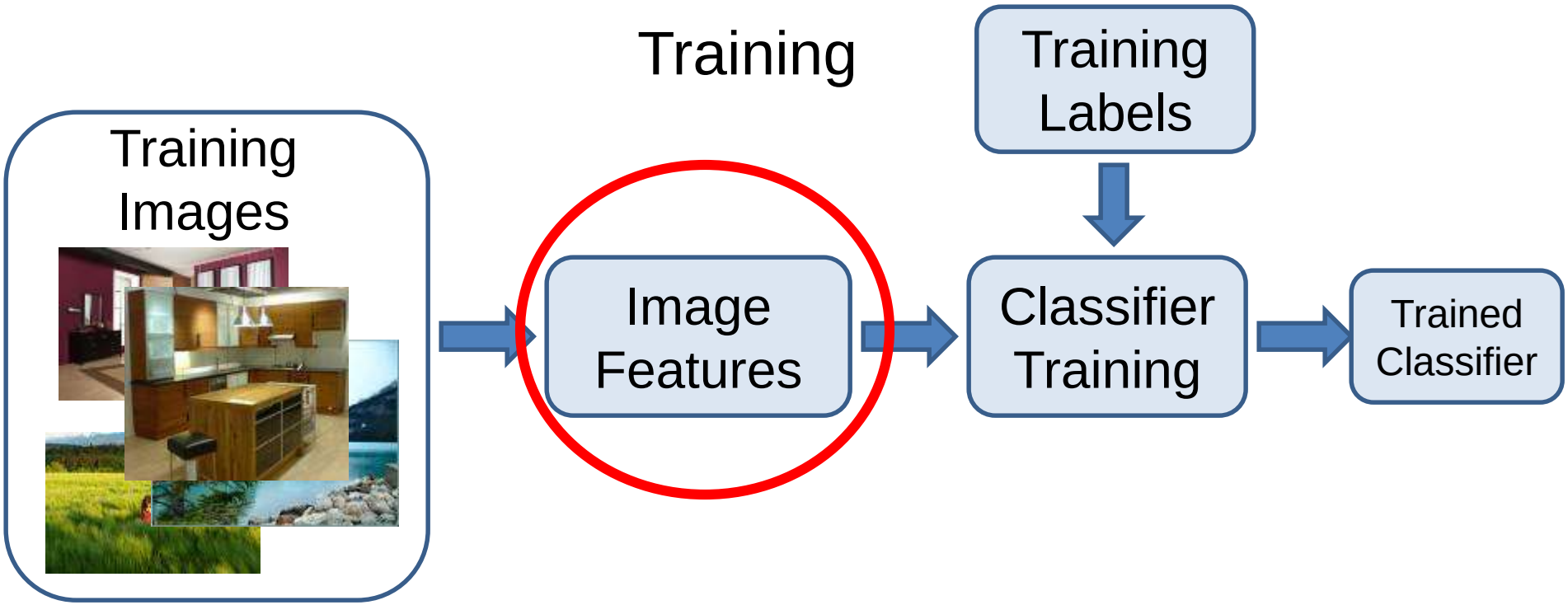
Image Features

Trained Classifier

Prediction

Outdoor

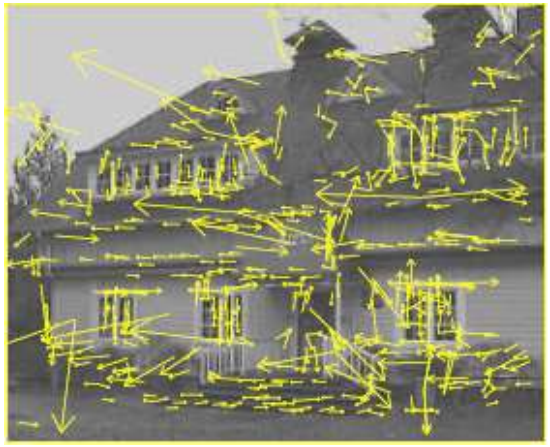
Part 1: Image features

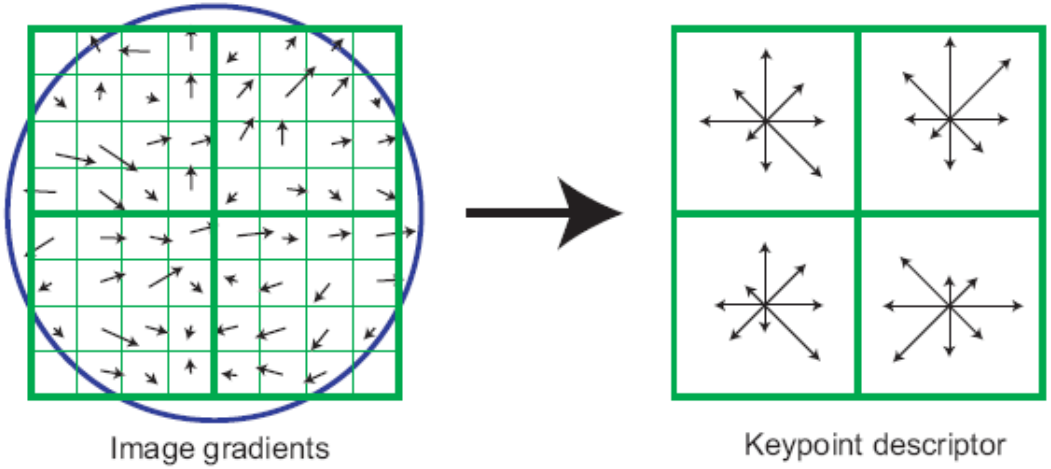


Histograms of Oriented Gradients

SIFT – Lowe IJCV 2004

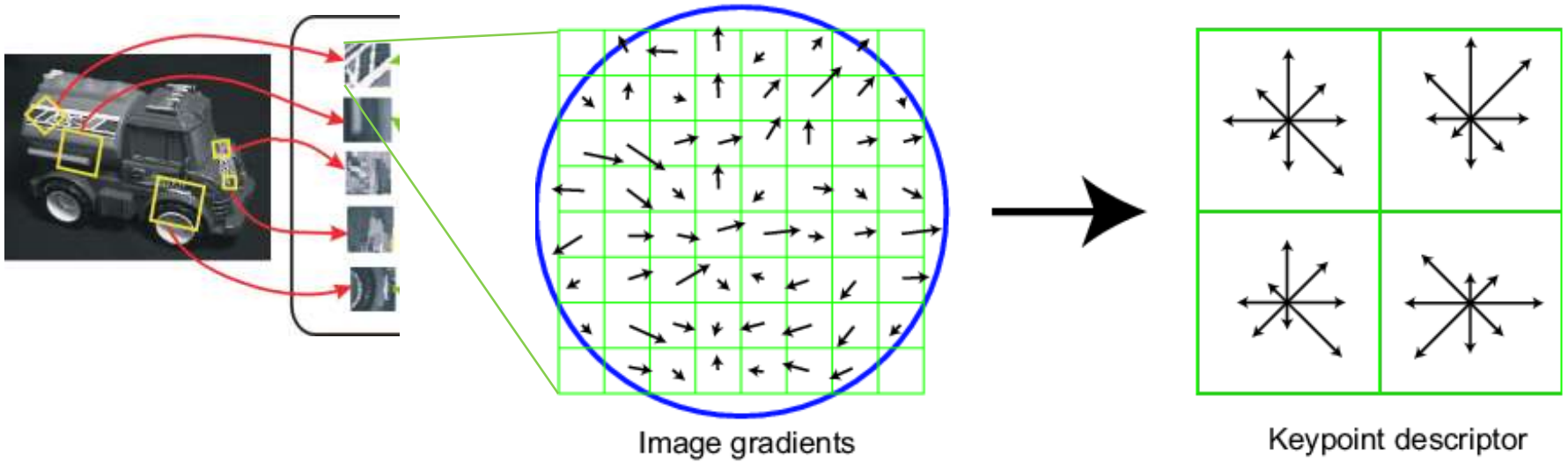
- Histograms of oriented gradients





SIFT Vector Formation

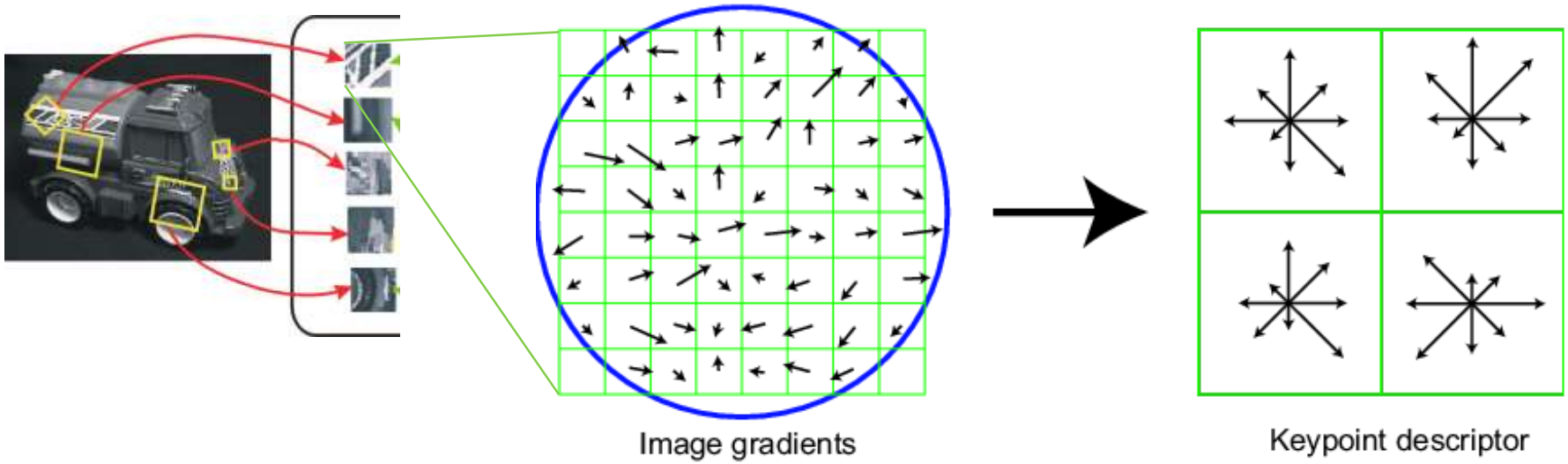
- 4x4 array of gradient orientation histograms
  - not really histogram, weighted by magnitude
- 8 orientations 4x4 array = 128 dimensions



showing only 2x2 here but is 4x4

Reduce Effect of Illumination

- 128-dim vector normalized to 1
- Threshold gradient magnitudes to avoid excessive influence of high gradients
  - after normalization, clamp gradients > 0.2
  - renormalize



Photometric Analysis: Recap

- Model images assuming Lambertian reflectance:

$$C = m_b(\mathbf{n}, \mathbf{s}) \int_{\lambda} e(\lambda)\rho(\lambda) f_C(\lambda) d\lambda$$

- Image transformation using von Kries model[1]:

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} \alpha & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & \gamma \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix}$$

(1) - J. von Kries. "Influence of adaptation on the effects produced by luminous stimuli." In D. MacAdam, editor, *Sources of Color Vision*, pages 109–119. MIT Press, 1970.

Photometric Analysis

Categorize common changes in image values **f** in the diagonal-offset model

1. Light intensity change (a = b = c)

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix}$$

Examples: shadows, shading

➘ *scale-invariant* w.r.t. light intensity

$I^c = a\, I^u$

Photometric Analysis

Photometric Analysis

2. Light intensity shift ($a = b = c = 1$; $o_1 = o_2 = o_3$)

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$$

Examples: object highlights under white light source, scattering of a white source

➘ *shift-invariant* w.r.t. light intensity

$I^c = I^u + o_1$

Photometric Analysis

Photometric Analysis

3. Light intensity change *and* shift (a = b = c; $o_1 = o_2 = o_3$)

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$$

➘ *scale-invariant* and *shift-invariant*

$I^c = a\, I^u + o_1$

Photometric Analysis

Light color change

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix}$$

Light color change and shift

$$\begin{pmatrix} R^c \\ G^c \\ B^c \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R^u \\ G^u \\ B^u \end{pmatrix} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix}$$

(1,1,1)

(1.8,1.2,1.2)

Color Descriptor Taxonomy

[van de Sande, IEEE PAMI, 2010]

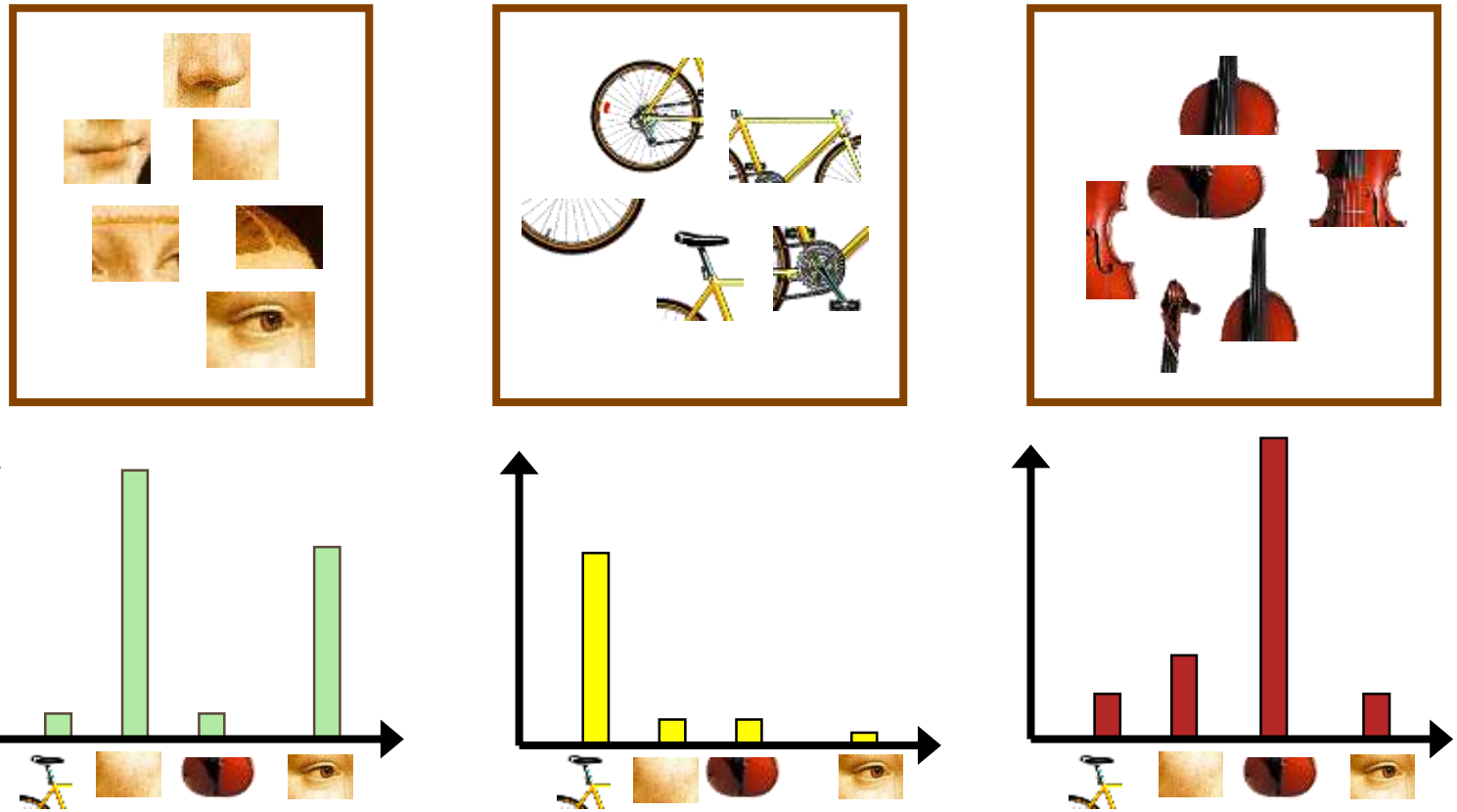
	Light intensity change $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light intensity shift $\begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light intensity change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light color change $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light color change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix}$
RGB Histogram	-	-	-	-	-
O_1, O_2	-	+	-	-	-
O_3, Intensity	-	-	-	-	-
Hue	+	+	+	-	-
Saturation	+	+	+	-	-
r, g	+	-	-	-	-
Transformed color	+	+	+	+	+
Color moments	-	+	-	-	-
Moment invariants	+	+	+	+	+
SIFT (∇I)	+	+	+	+	+
HSV-SIFT	+	+	+	+/-	+/-
HueSIFT	+	+	+	+/-	+/-
OpponentSIFT	+/-	+	+/-	+/-	+/-
W-SIFT	+	+	+	+/-	+/-
rgSIFT	+	+	+	+/-	+/-
Transf. color SIFT	+	+	+	+	+

Bag-of-Words Model

Svetlana Lazebnik

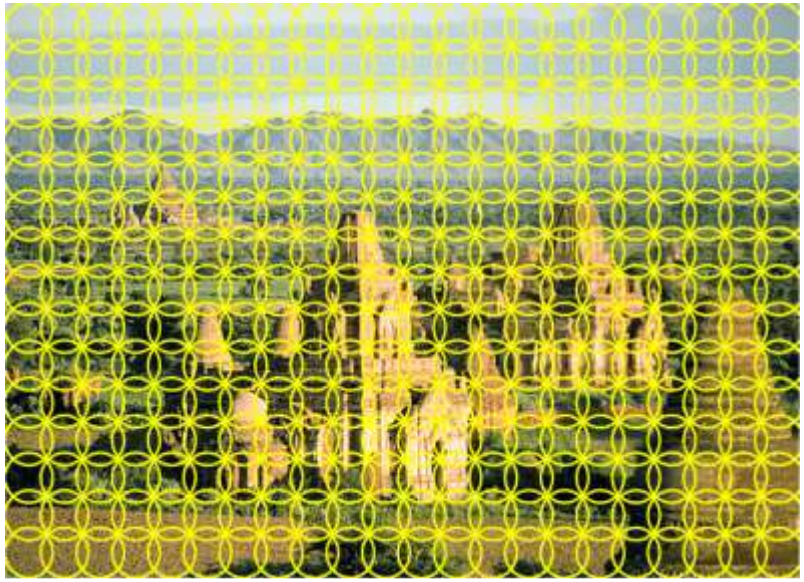
Bag-of-Features Steps

1. Extract features
2. Learn “visual vocabulary”
3. Quantize features using visual vocabulary
4. Represent images by frequencies of “visual words”

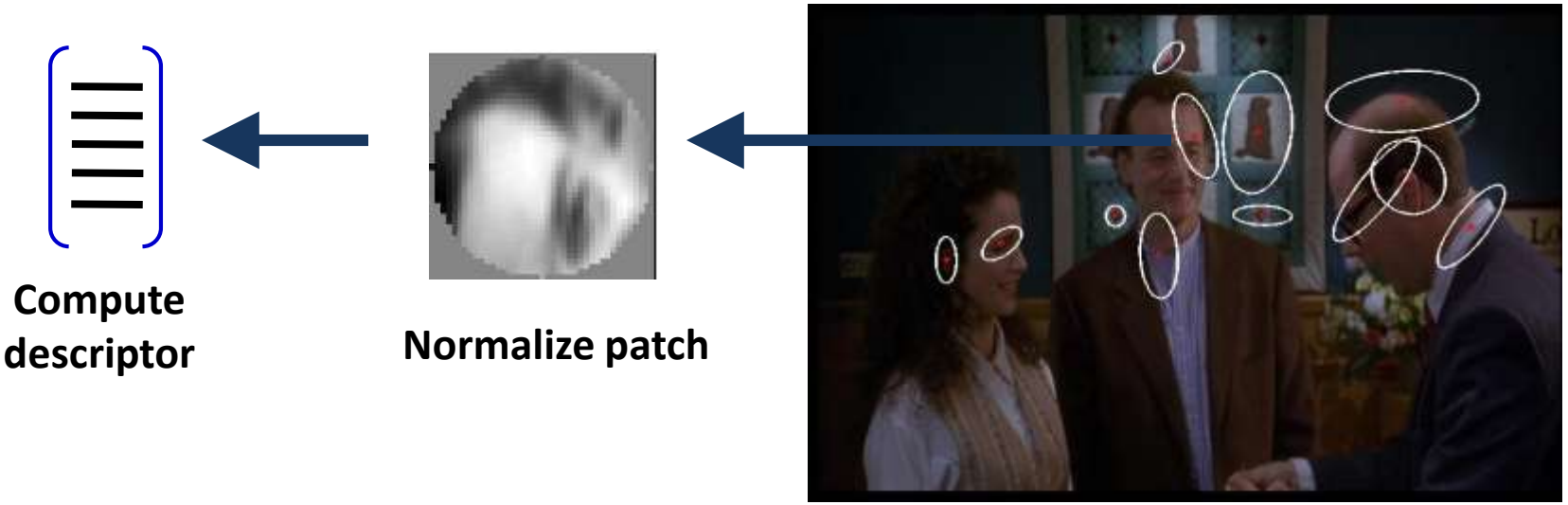


1. Feature extraction

- Regular grid or interest regions

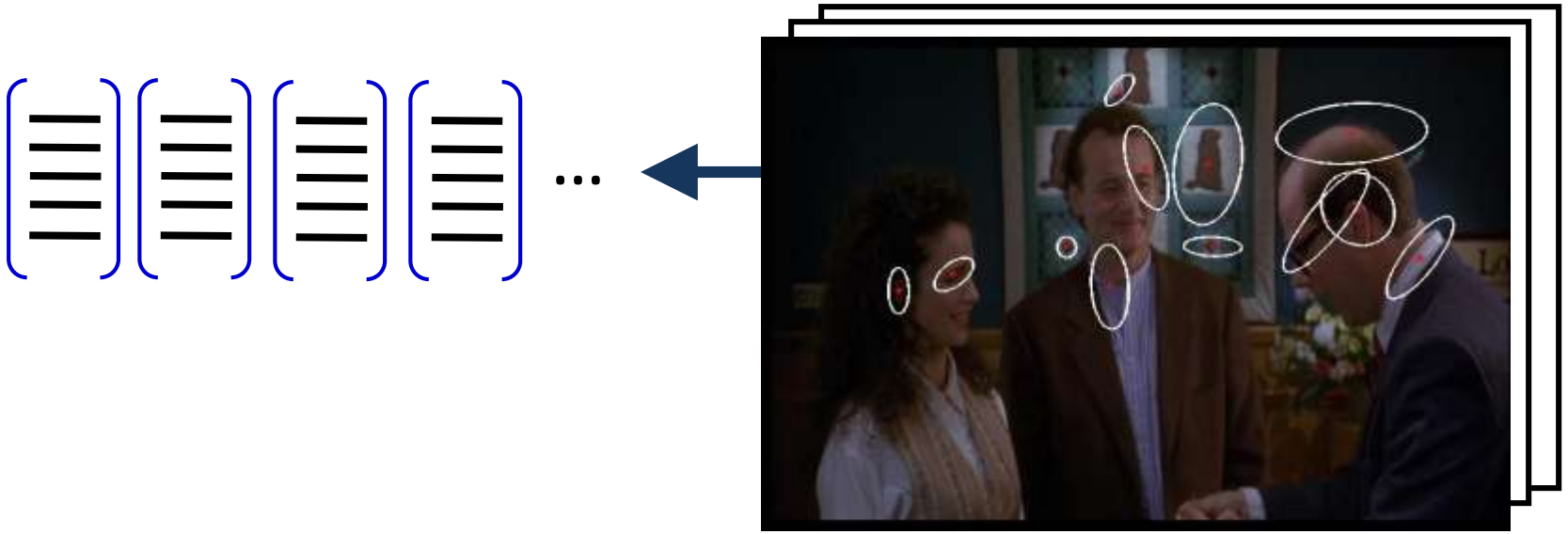


1. Feature extraction



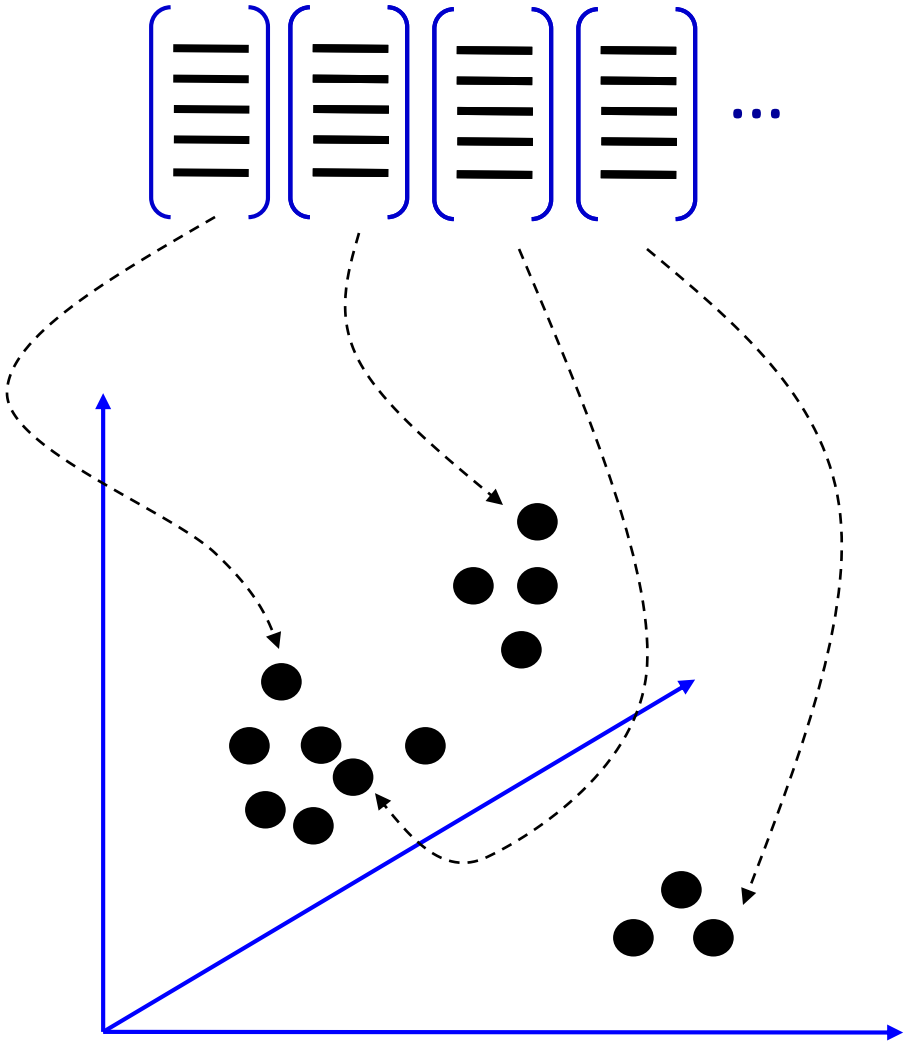
Slide credit: Josef Sivic

1. Feature extraction



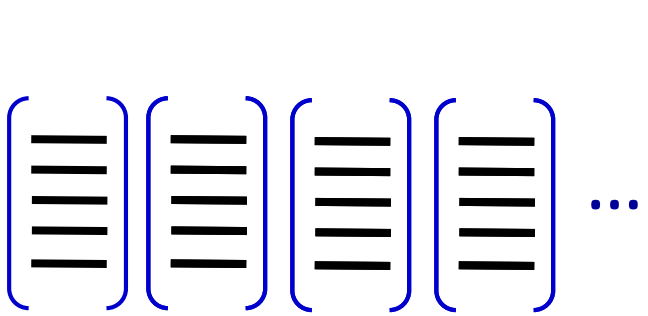
Slide credit: Josef Sivic

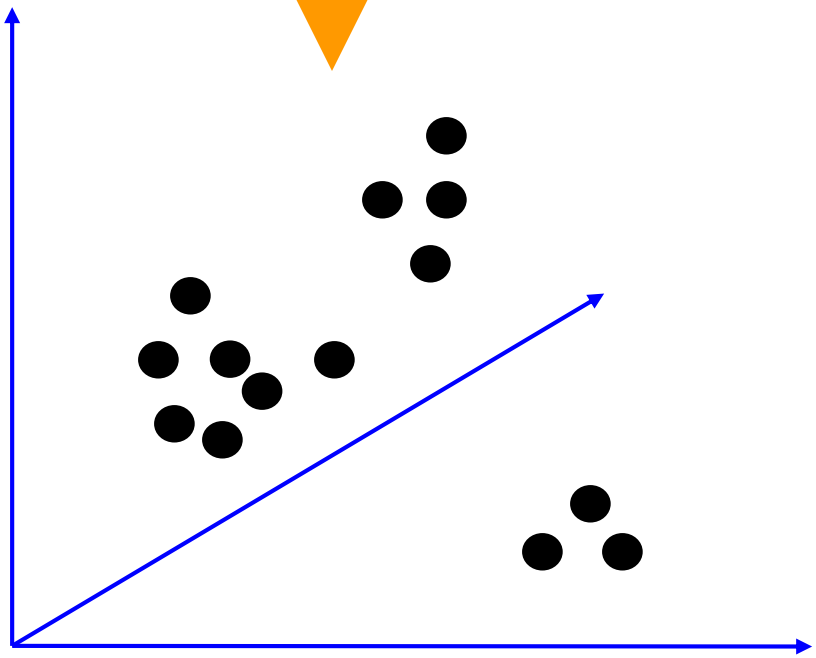
2. Learning the visual vocabulary

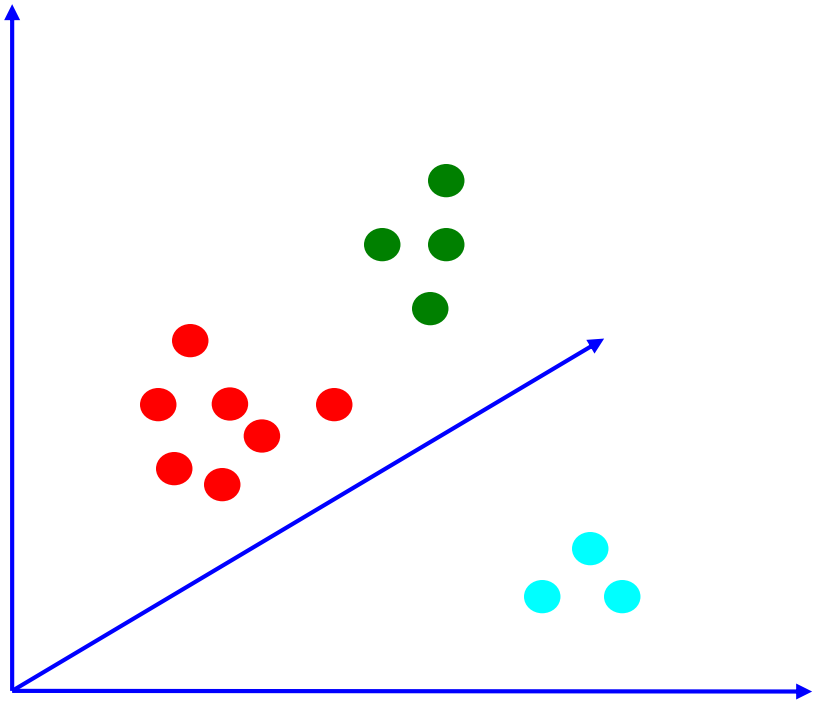


Slide credit: Josef Sivic

2. Learning the visual vocabulary



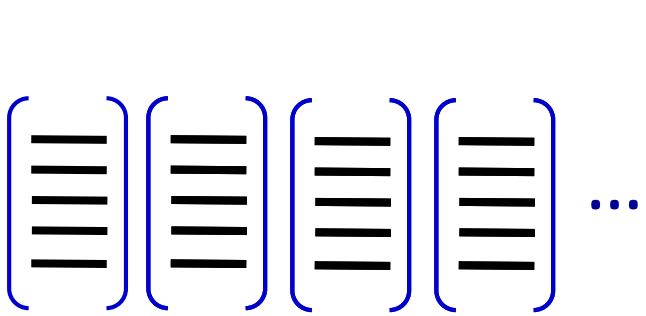


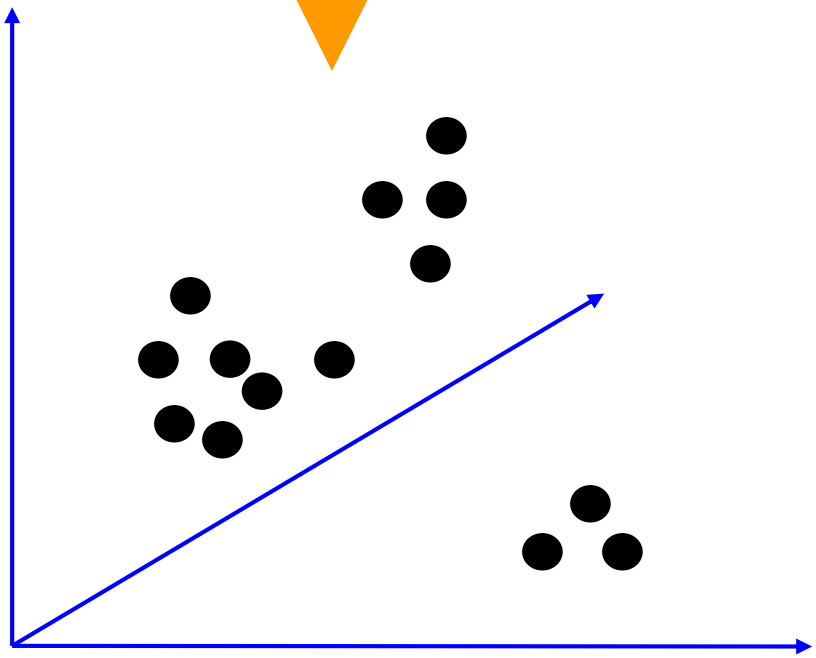


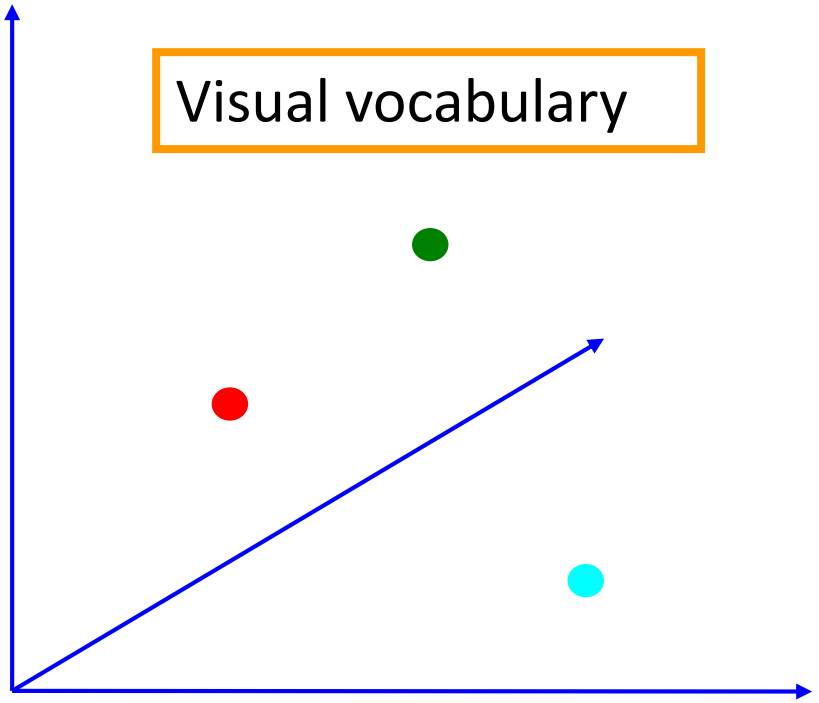
Clustering

Slide credit: Josef Sivic

2. Learning the visual vocabulary







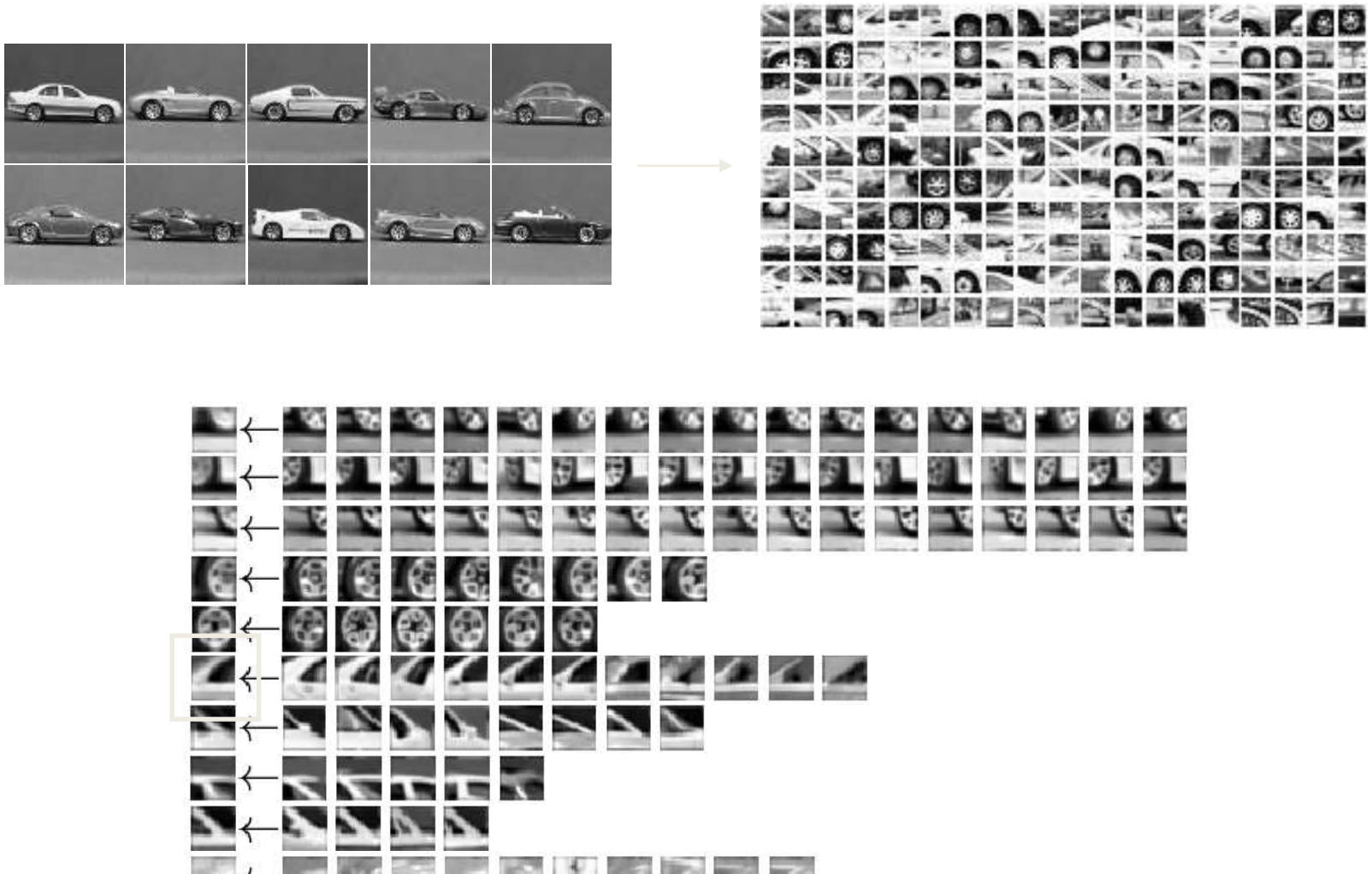
Slide credit: Josef Sivic

Clustering and Vector Quantization

- Clustering is a common method for learning a visual vocabulary or codebook
  - Unsupervised learning process
  - Each cluster center produced by k-means becomes a codevector
  - Codebook can be learned on separate training set
  - Provided the training set is sufficiently representative, the codebook will be “universal”

- The codebook is used for quantizing features
  - A *vector quantizer* takes a feature vector and maps it to the index of the nearest codevector in a codebook
  - Codebook = visual vocabulary
  - Codevector = visual word

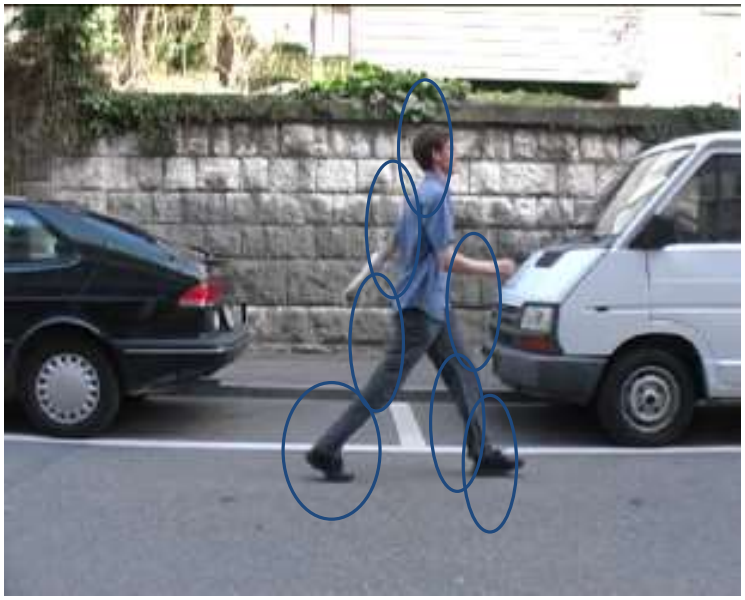
Example Codebook

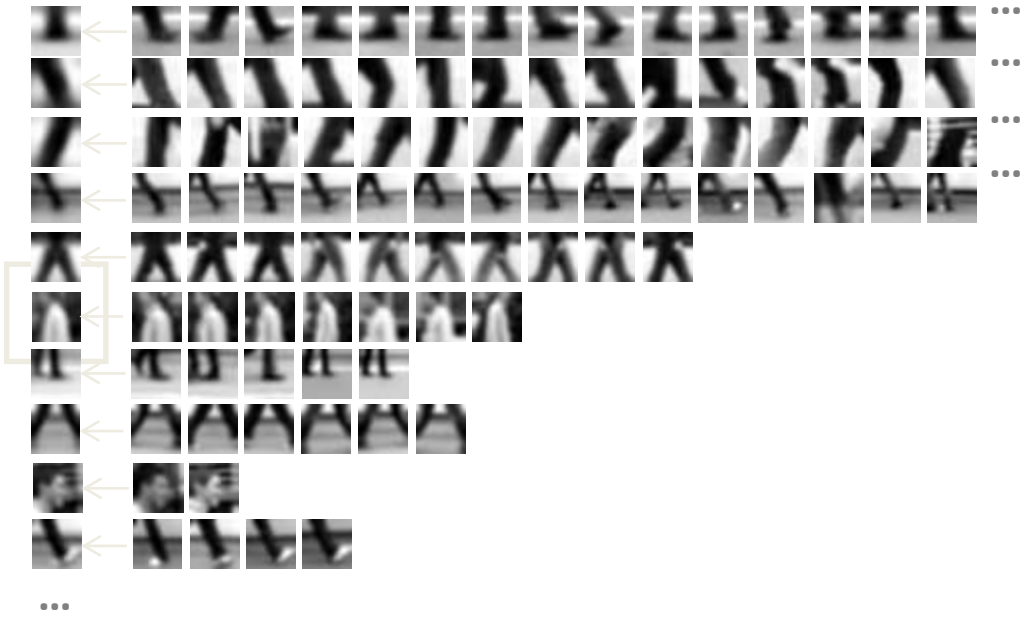


Appearance codebook

Source: B. Leibe

Another Codebook

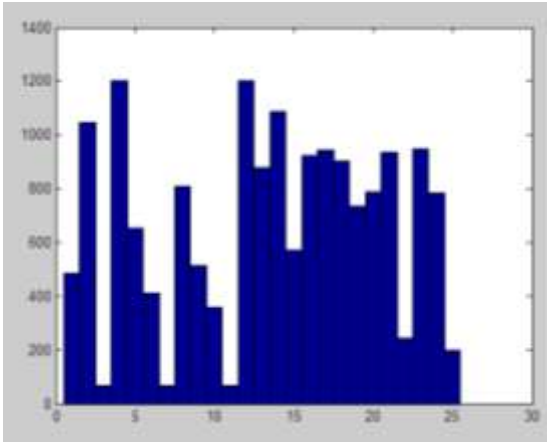




Appearance codebook

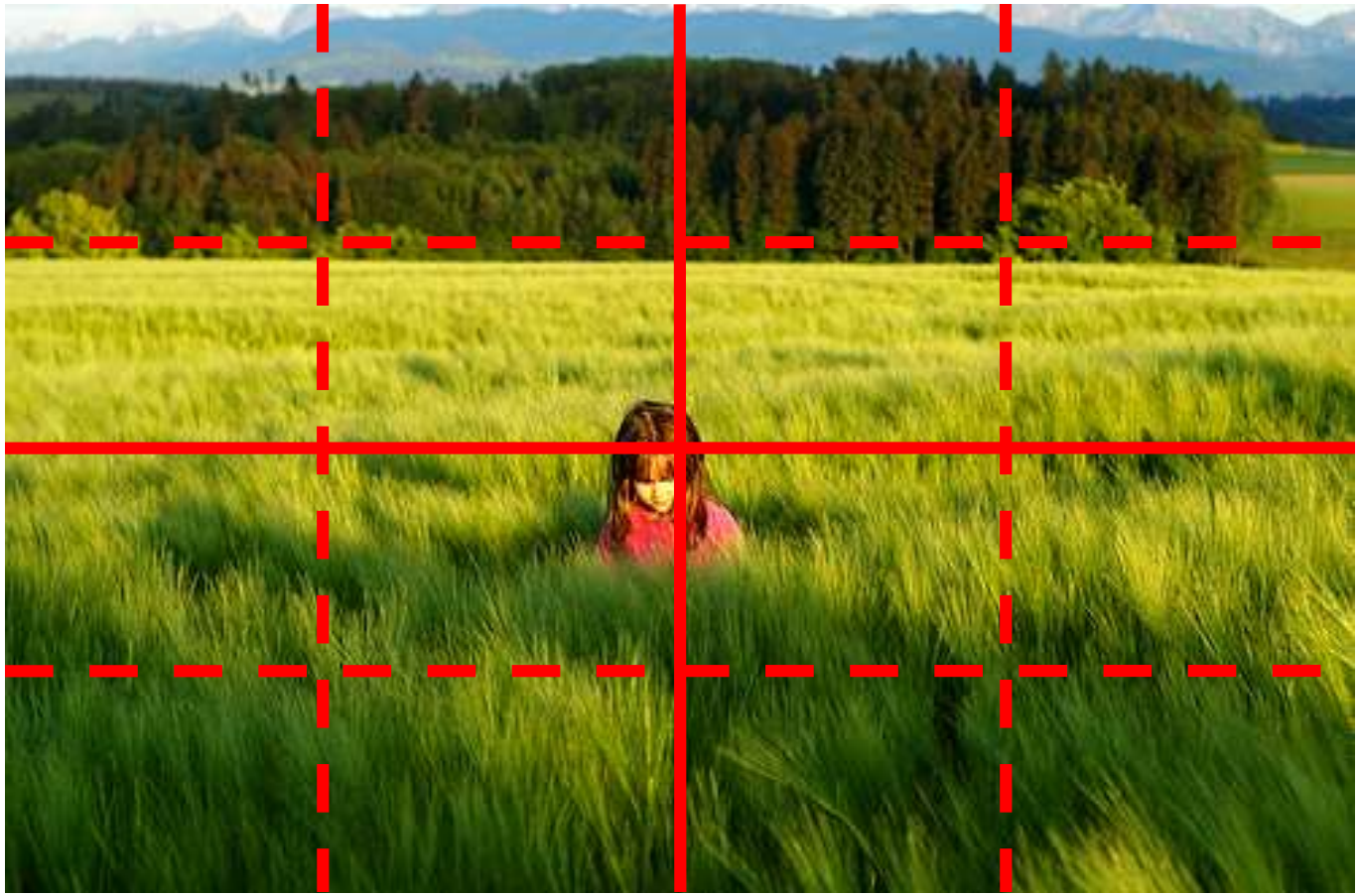
Source: B. Leibe

But What About Layout?



All of these images have the same color histogram

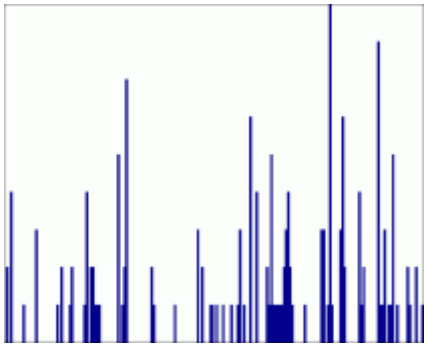
Spatial Pyramid



Compute histogram in each spatial bin

Spatial Pyramid Representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution

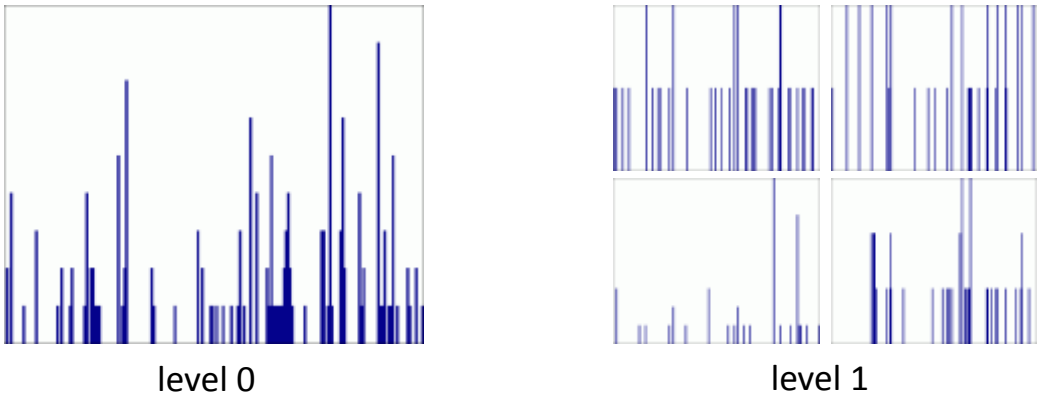


level 0

Lazebnik, Schmid & Ponce (CVPR 2006)

Spatial Pyramid Representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution

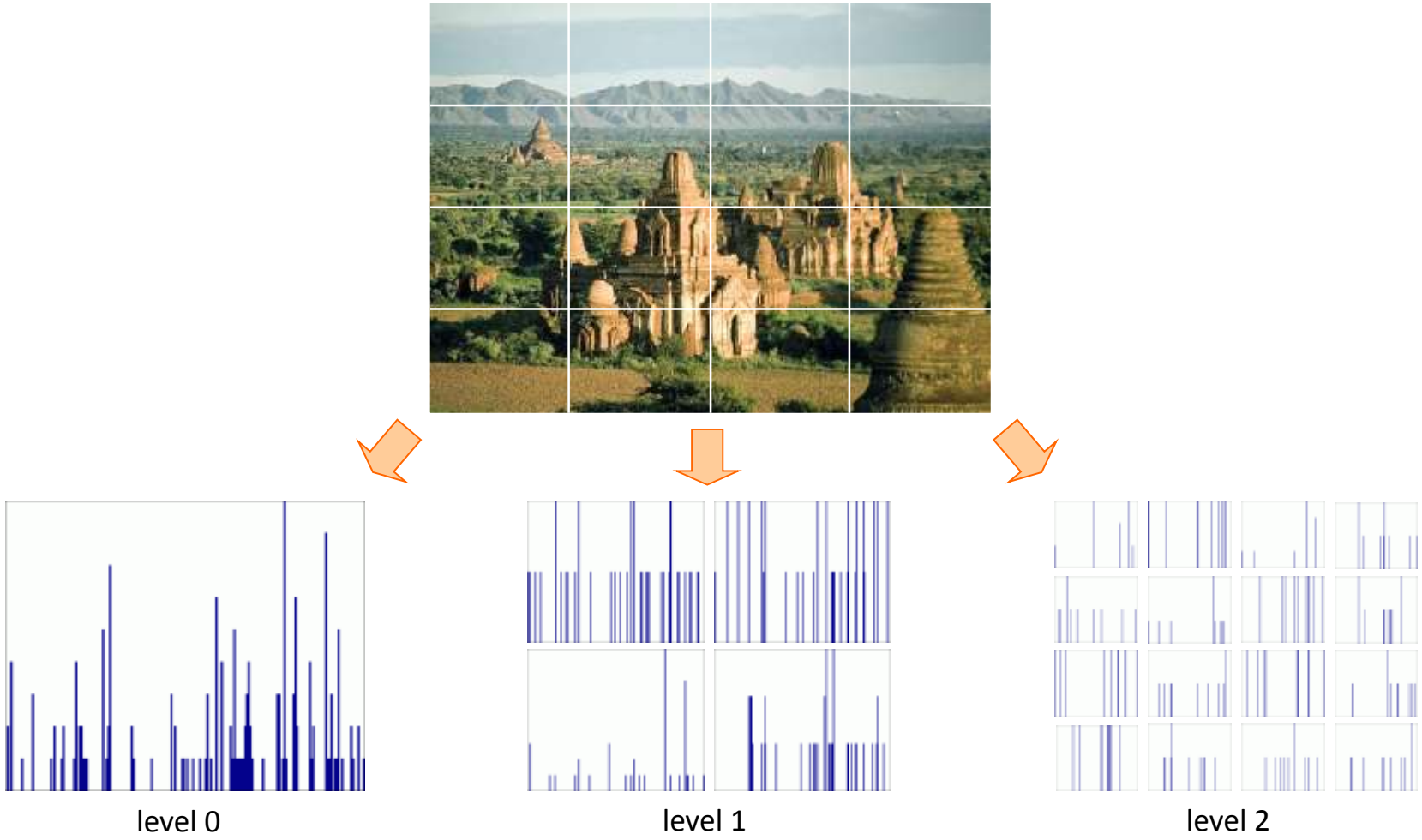


level 0 level 1

Lazebnik, Schmid & Ponce (CVPR 2006)

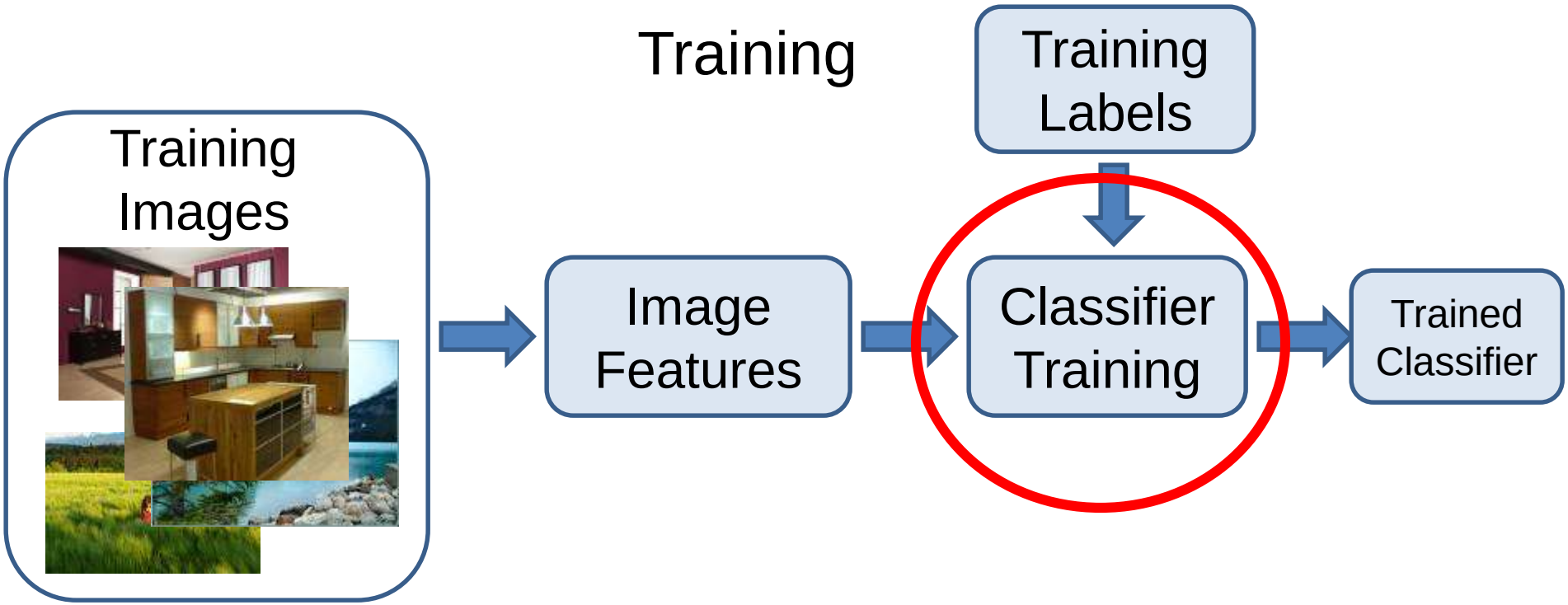
Spatial Pyramid Representation

- Extension of a bag of features
- Locally orderless representation at several levels of resolution



Lazebnik, Schmid & Ponce (CVPR 2006)

Classifiers

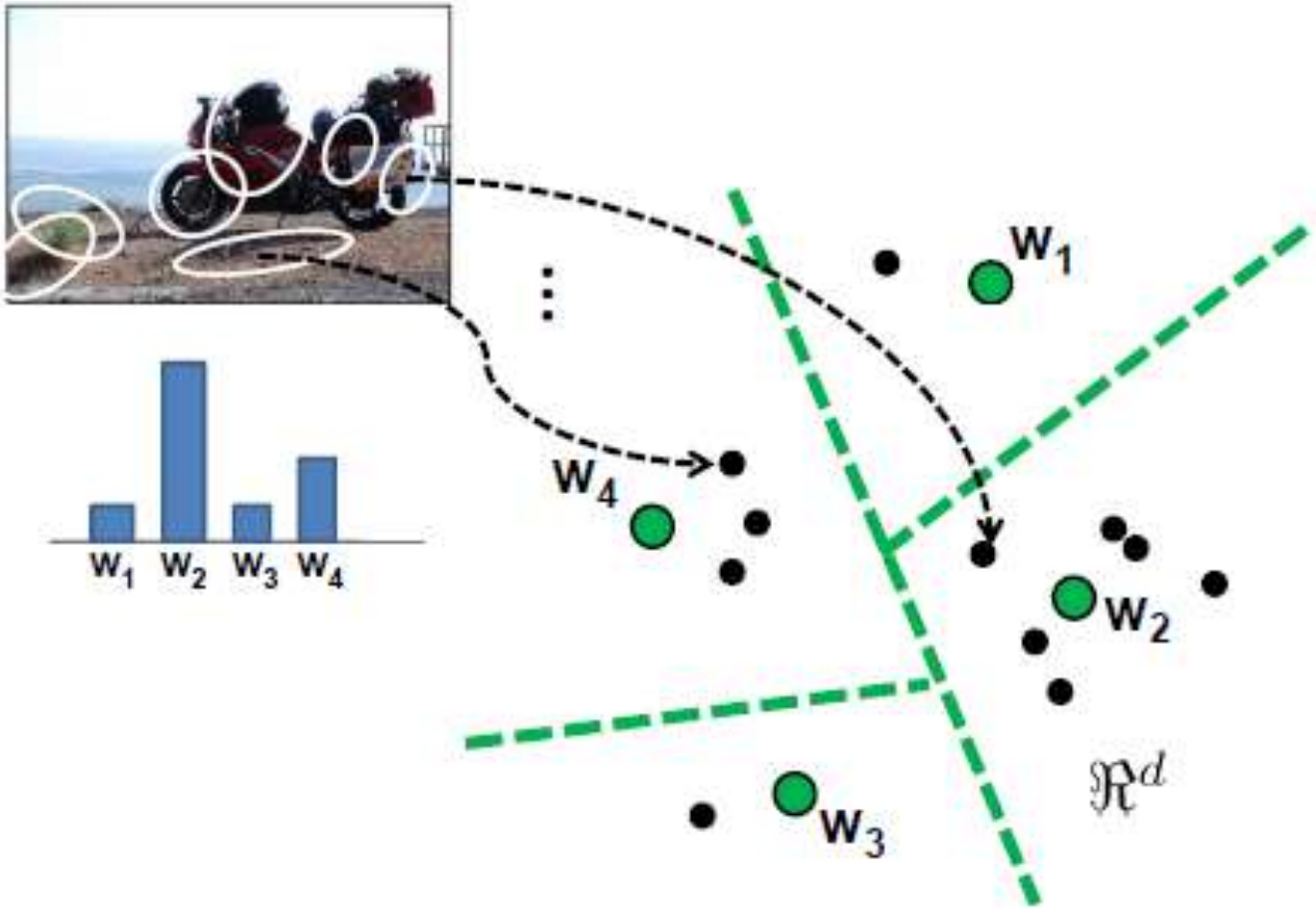


Slide: Derek Hoiem

BoW and codebook encoding or vector quantization

A vector quantizer takes a feature vector
and maps it to the index of the nearest codevector in a codebook

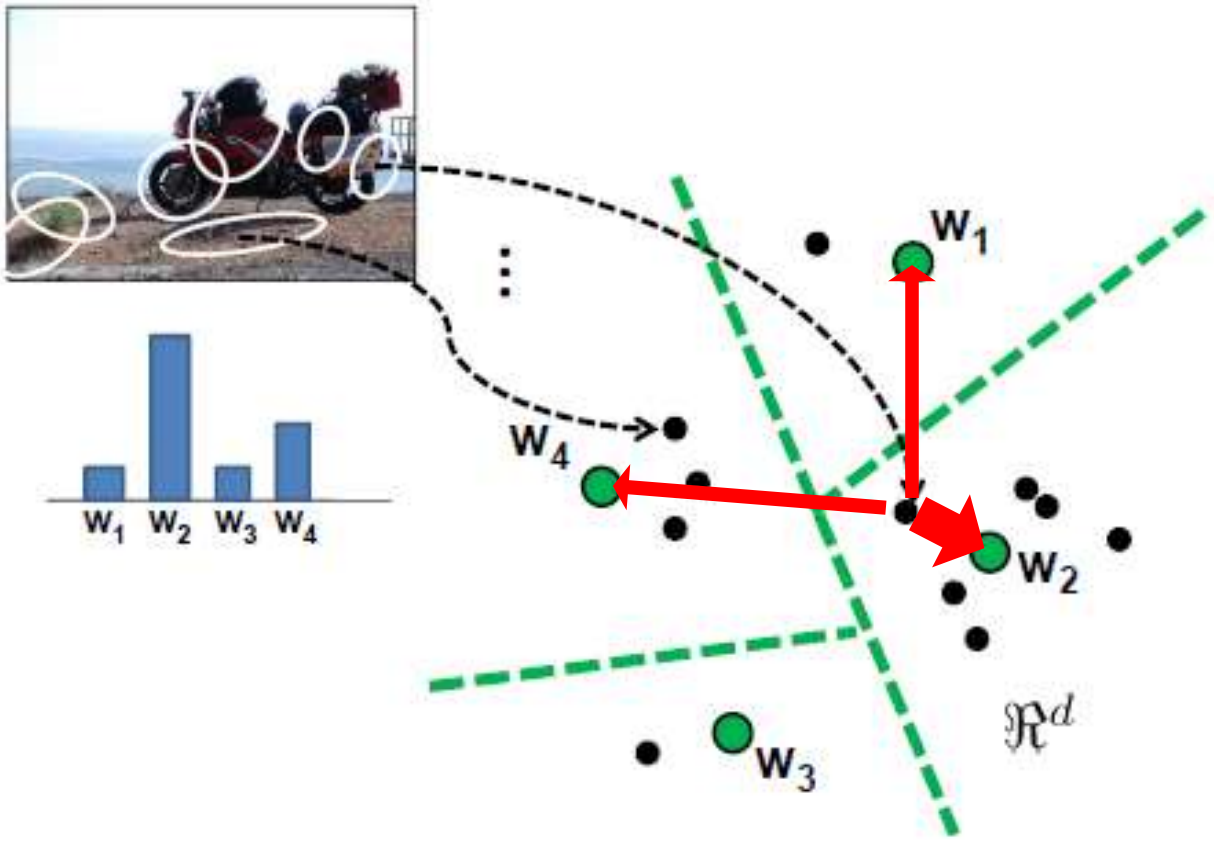
Hard Assignment



http://www.cs.utexas.edu/~grauman/courses/fall2009/papers/bag_of_visual_words.pdf

Soft Assignment

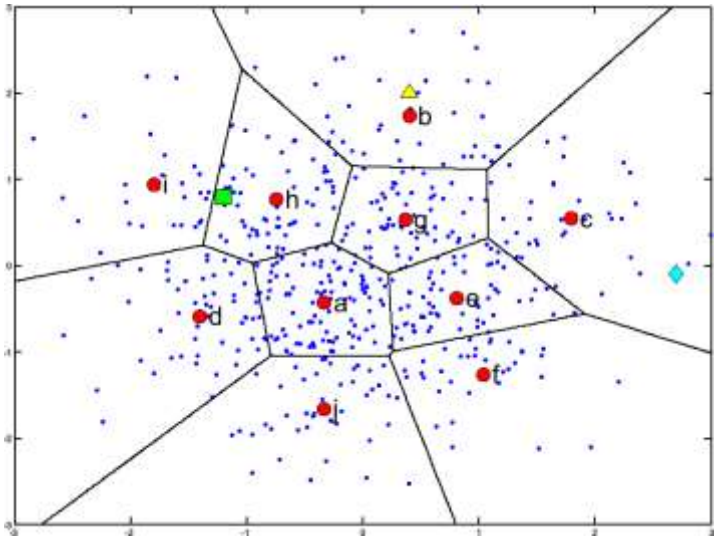
- Called “Kernel codebook encoding” by Chatfield et al. 2011. Cast a weighted vote into the most similar clusters.



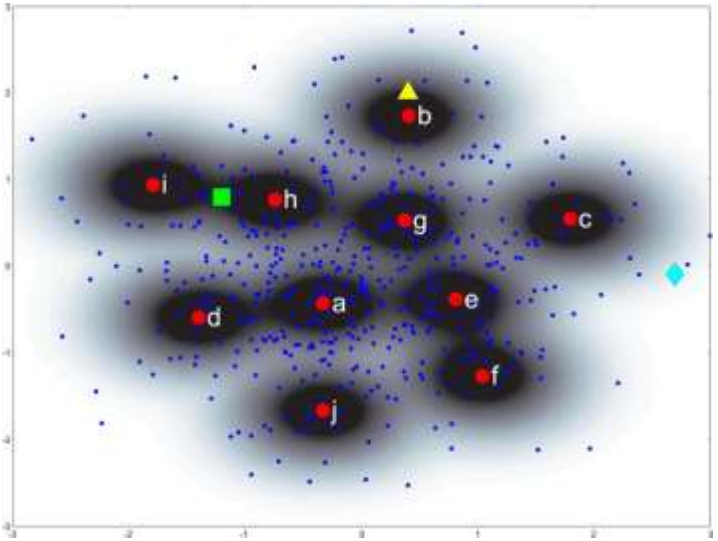
Codebook Assignment

Soft assignment using Gaussian kernel

● Codeword



Hard assignment



Soft assignment

Assignment	MAP on TV2007test
Hard	0,155
Soft	0,166

relative +7%

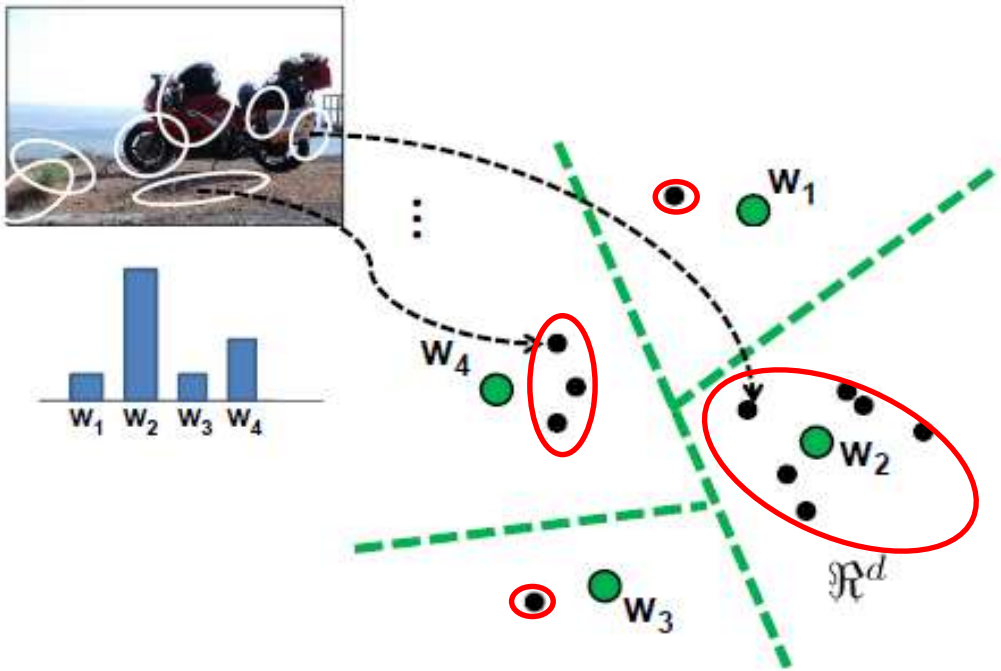
[van Gemert, ECCV08]

Motivation

Bag of Visual Words is only about **counting** the number of local descriptors assigned to each Voronoi region

Why not including **other statistics**? For instance:

- mean of local descriptors
- (co)variance of local descriptors

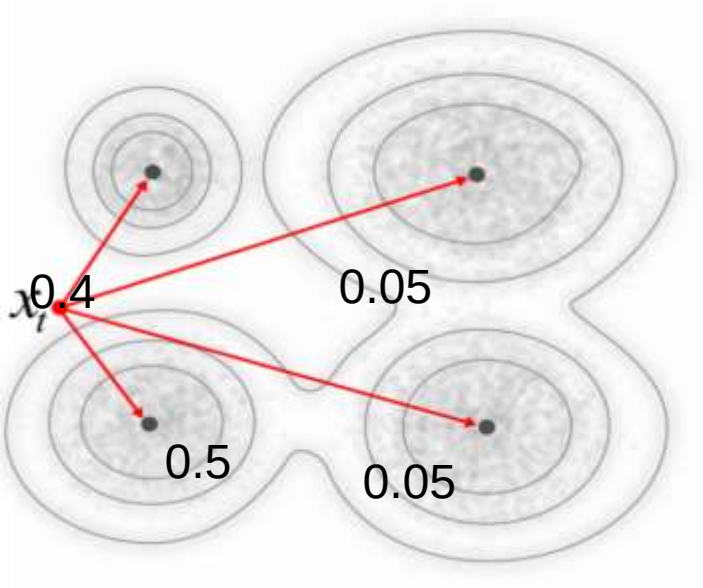


F. Perronnin, H. Jégou, CVPR tutorial on LSVRC, June 16, 2012.

http://www.cs.utexas.edu/~grauman/courses/fall2009/papers/bag_of_visual_words.pdf

Fisher Vector and GMM

- For Fisher Vector image representations, u_λ is a GMM.
- GMM can be thought of as “soft” kmeans.

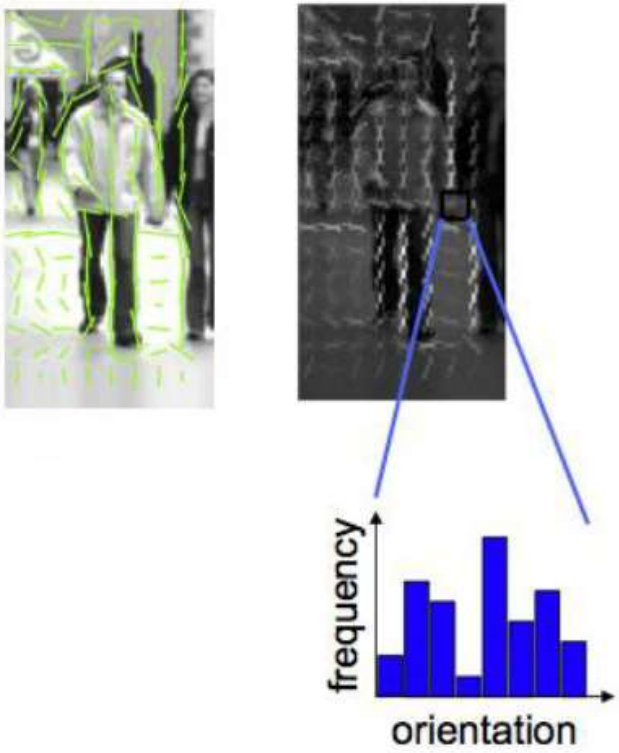


- Each component has a mean and a standard deviation along each direction (or full covariance)

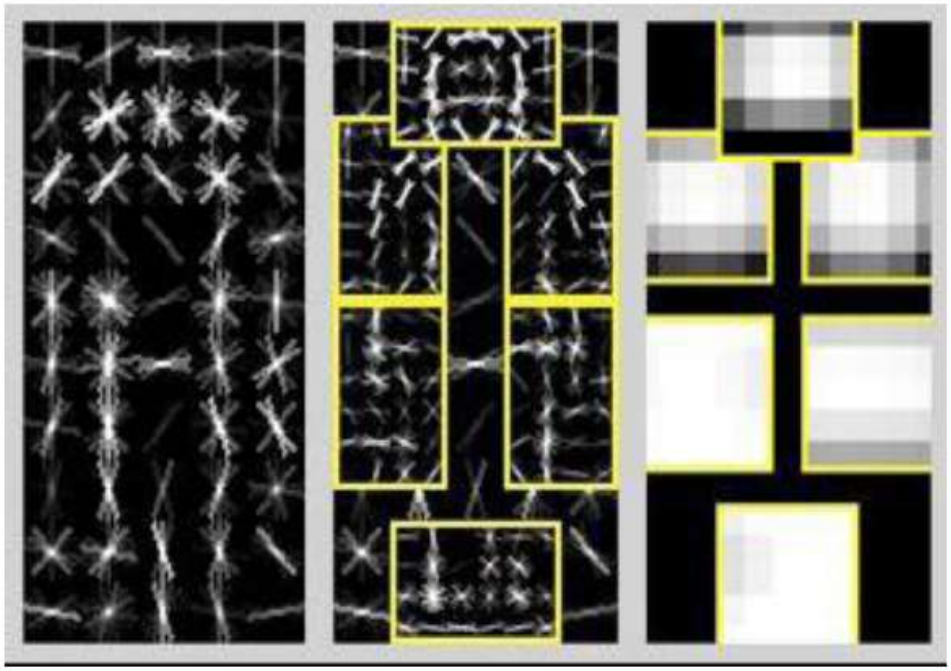
Codebook Summary

- We've looked at methods to better characterize the distribution of visual words in an image:
  - Soft assignment (a.k.a. Kernel Codebook)
  - Fisher Vector

- Mixtures of Gaussians is conceptually a soft form of kmeans which can better model the data distribution.



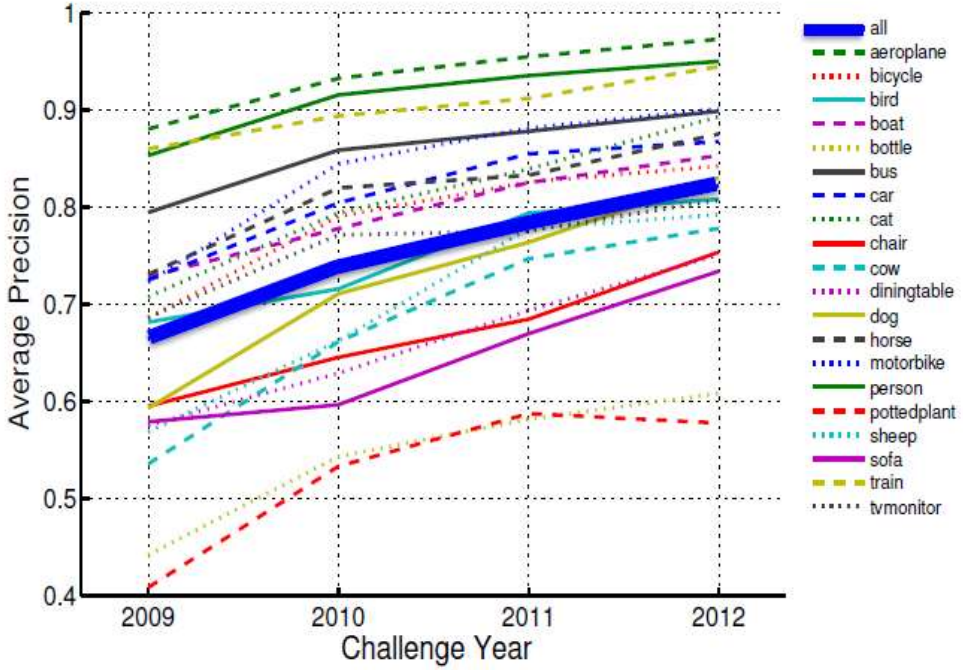
Histogram of Gradients (HoG)
Dalal & Triggs, 2005



Deformable Part Model
Felzenswalb, McAllester, Ramanan, 2009

PASCAL Visual Object Challenge (20 object categories)

[Everingham et al. 2006-2012]



IMAGENET

www.image-net.org

22K categories and **14M** images

- Animals
  - Bird
  - Fish
  - Mammal
  - Invertebrate
- Plants
  - Tree
  - Flower
- Food
- Materials
- Structures
- Artifact
  - Tools
  - Appliances
  - Structures
- Person
- Scenes
  - Indoor
  - Geological Formations
- Sport Activities

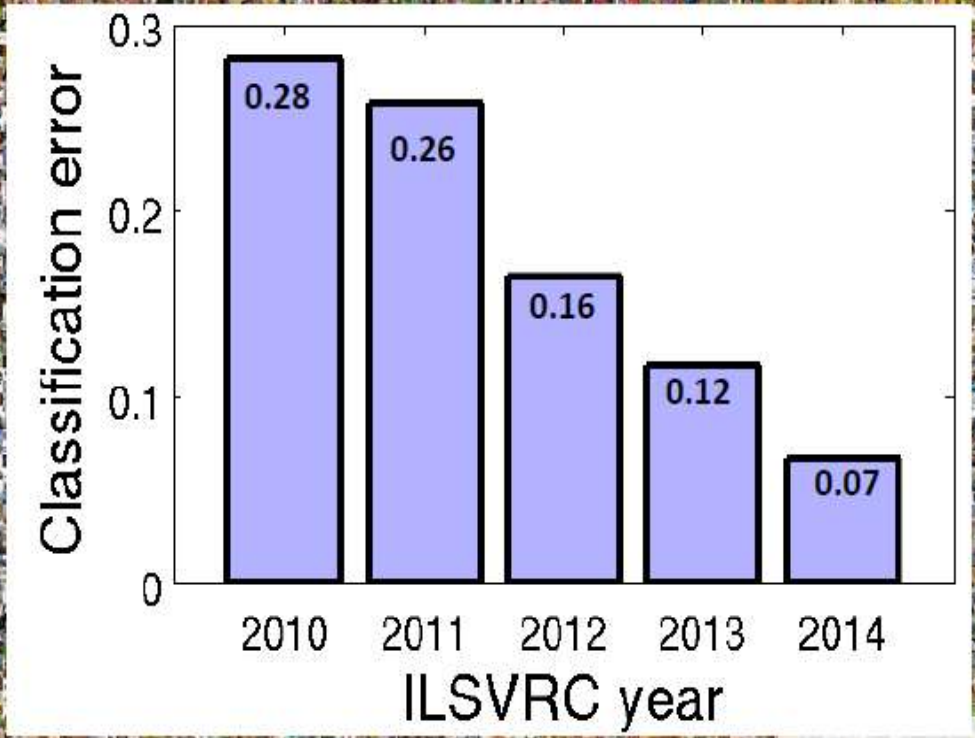
Deng, Dong, Socher, Li, Li, & Fei-Fei, 2009

IMAGENET Large Scale Visual Recognition Challenge

The Image Classification Challenge:

1,000 object classes

1,431,167 images



Russakovsky et al. arXiv, 2014

IMAGENET Large Scale Visual Recognition Challenge

Year 2010

NEC-UIUC

Dense grid descriptor: HOG, LBP

Coding: local coordinate, super-vector

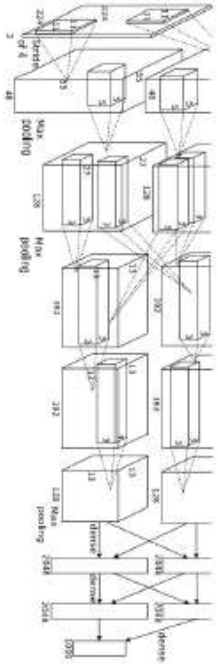
Pooling, SPM

Linear SVM

[Lin CVPR 2011]

Year 2012

SuperVision



[Krizhevsky NIPS 2012]

Year 2014

GoogLeNet

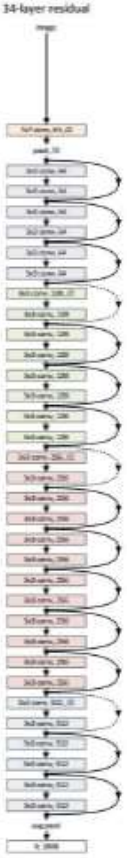
[Szegedy arxiv 2014]

VGG

[Simonyan arxiv 2014]

Year 2015

MSRA



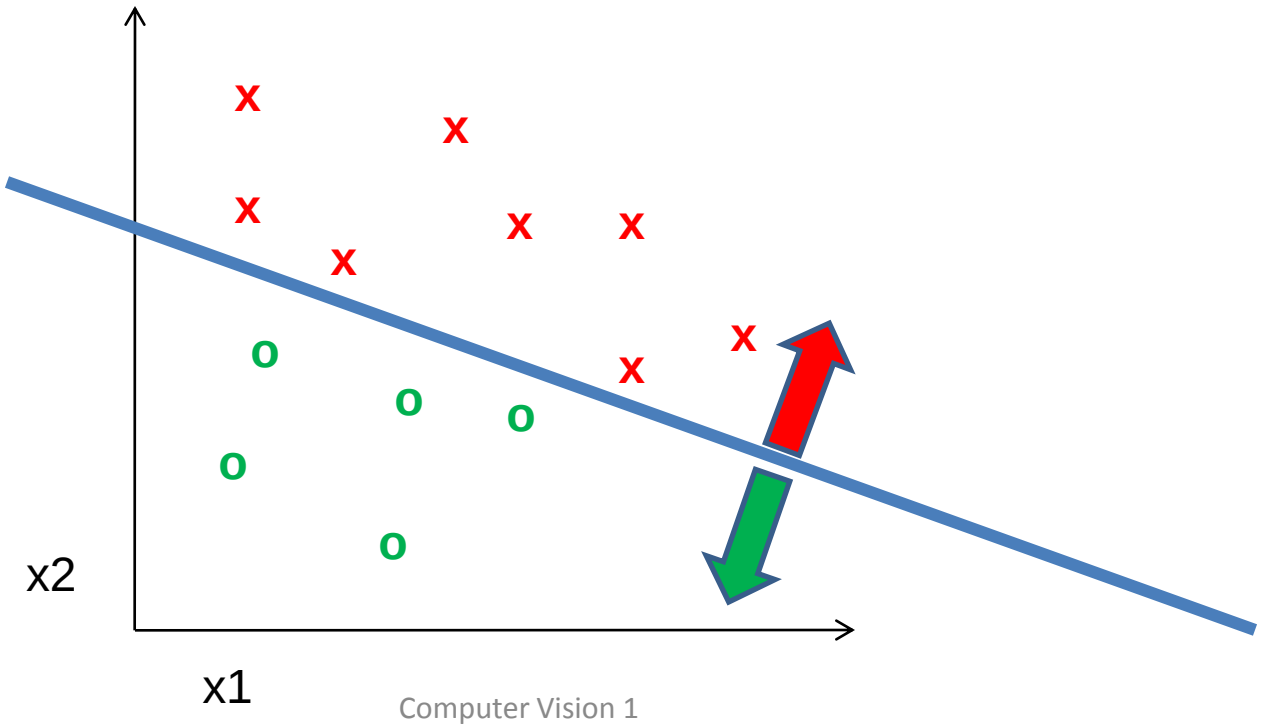
Many Classifiers to Choose From

- SVM
- Neural networks
- Naïve Bayes
- Bayesian network
- Logistic regression
- Randomized Forests
- Boosted Decision Trees
- K-nearest neighbor
- RBMs
- Etc.

Slide: Derek Hoiem

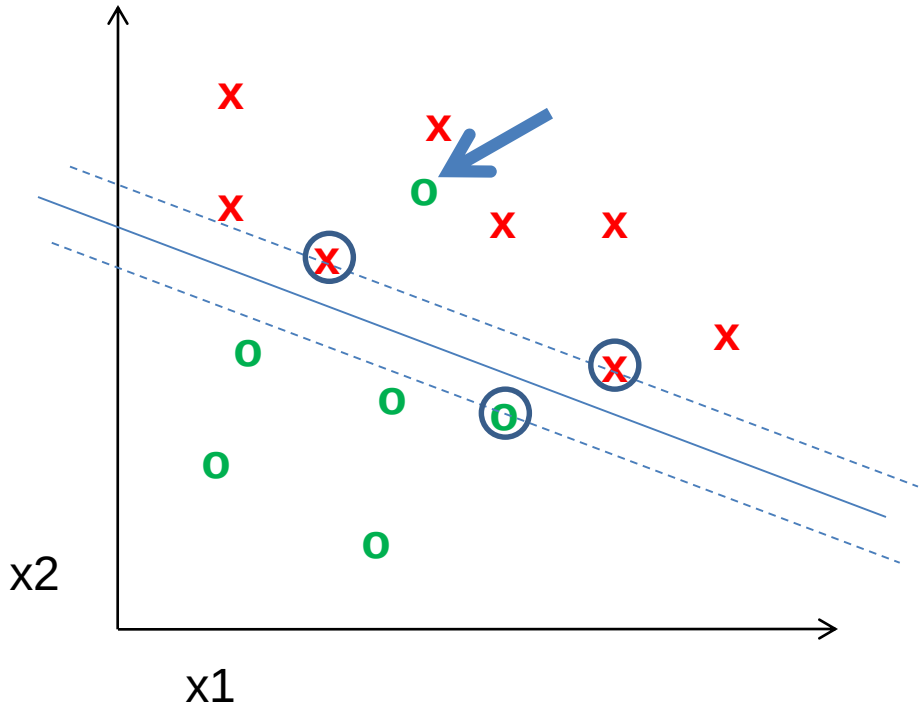
Learning a Classifier

Given some set of features with corresponding labels, learn a function to predict the labels from the features



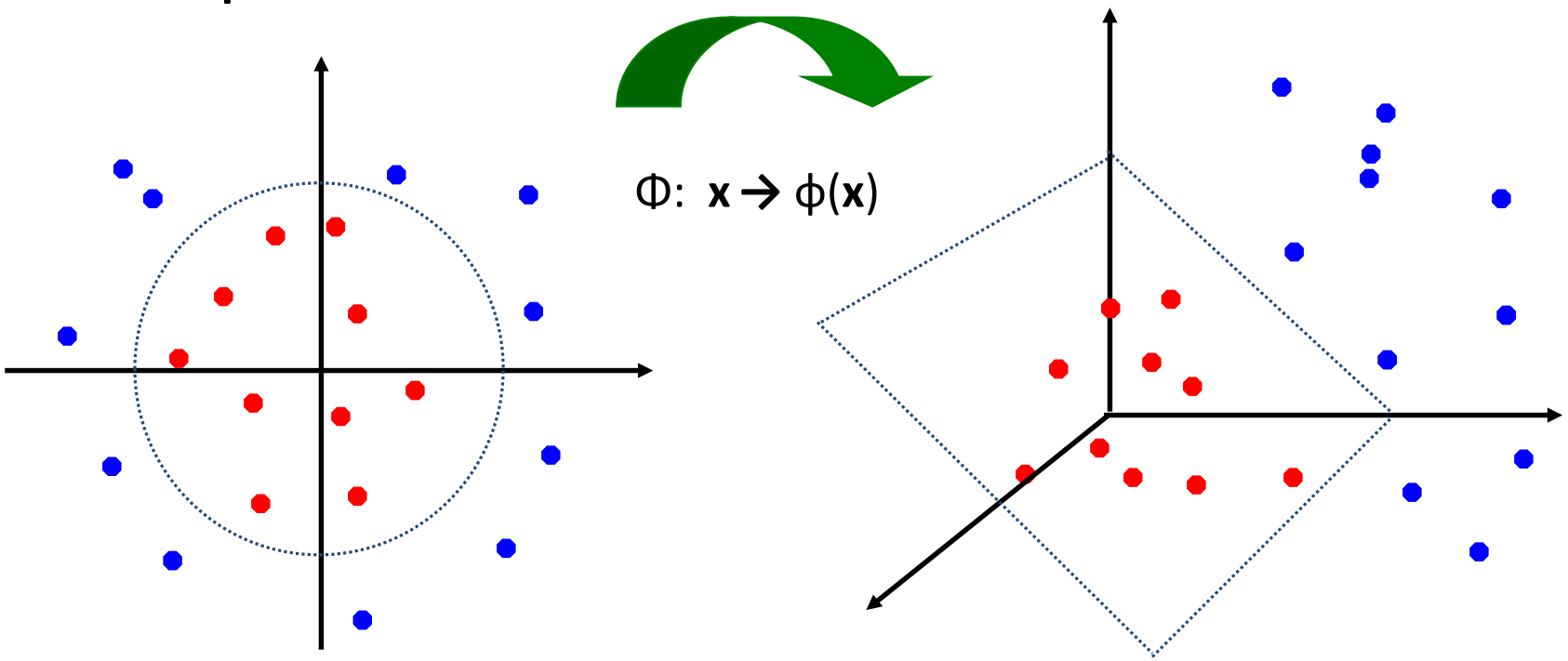
Slide: Derek Hoiem

Classifiers: Linear SVM



Nonlinear SVMs

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:



Slide credit: Andrew Moore

Design good classifiers

Example: Support Vector Machine (SVM) classifier

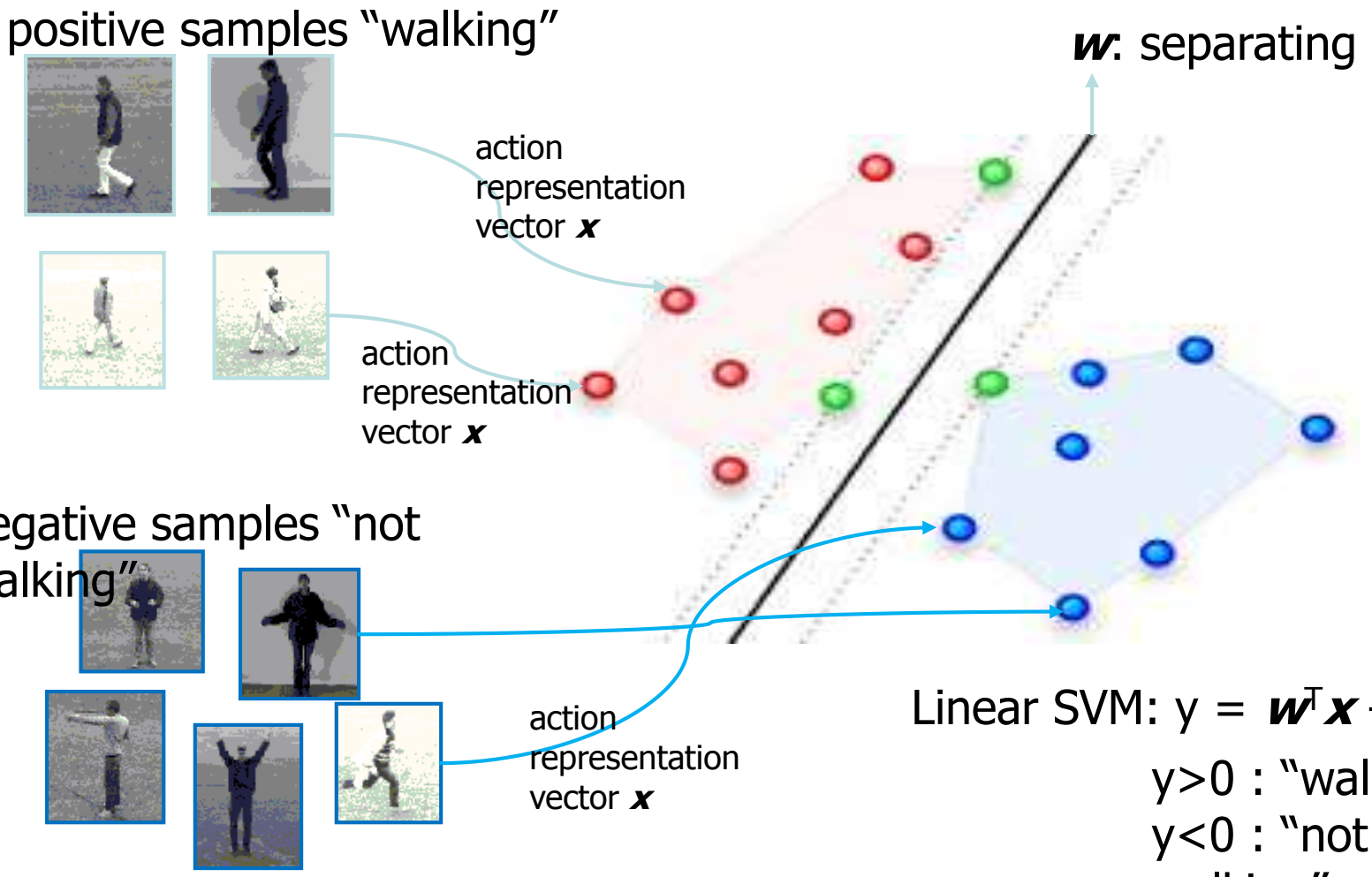


Image Categorization

Training

Training Images

Training Labels

Image Features

Classifier Training

Trained Classifier

Testing

Test Image

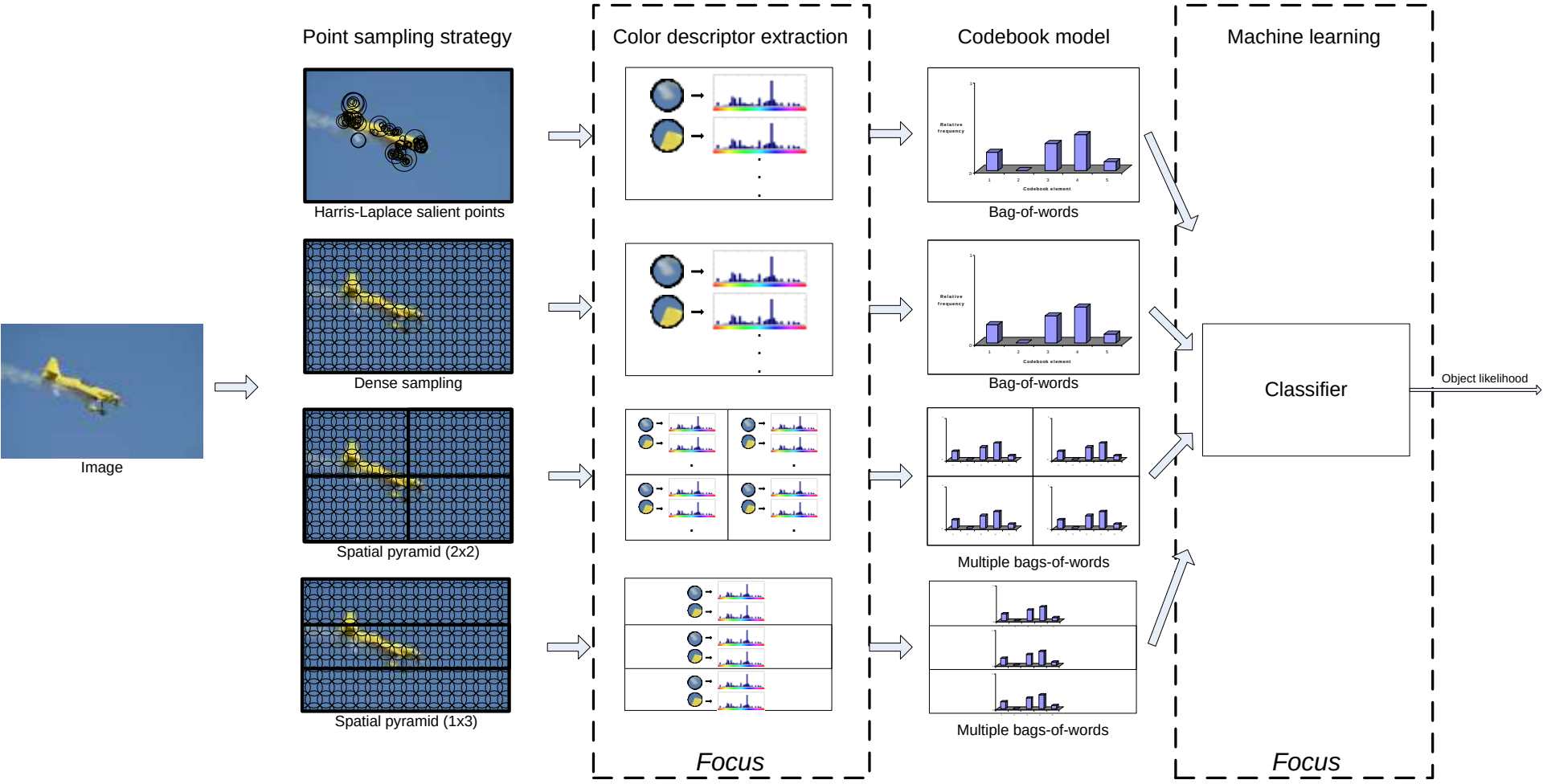
Image Features

Trained Classifier

Prediction

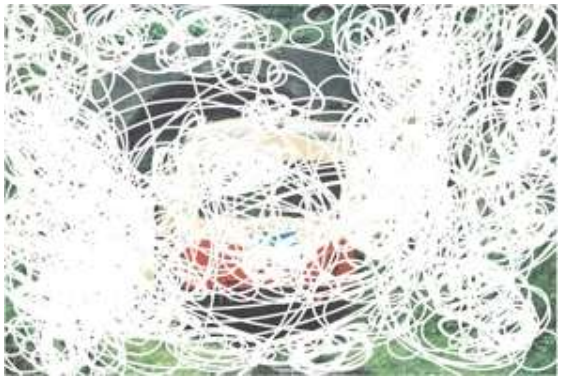
Outdoor

Pipeline Overview



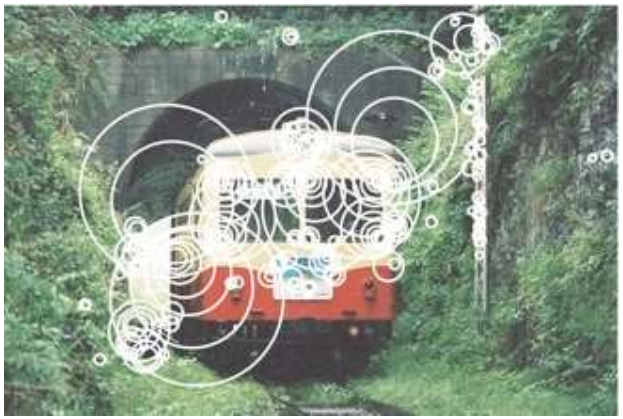
Original Image

Harris Laplacian
impl. by Mikolajczyk (e.g. CVPR06)



Shape adapted Harris Laplacian
impl. by Mikolajczyk (ICCV07)

Color salient points
Quasi invariant HSI



Color salient points
Color boosted OCS

Most of the time, both color approaches agree on the most salient parts of an image.

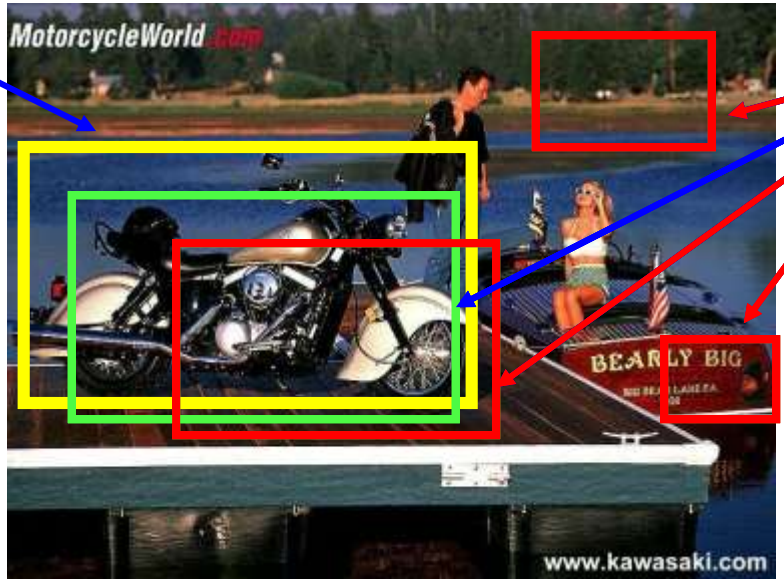
Distinctiveness

Distinctiveness studied experimentally:

- Image benchmark: PASCAL Visual Object Classes Challenge 2007
- 9963 photos from Flickr
- 20 object types
- Earth Movers Distance (EMD) between cluster sets of different images, used in EMD kernel function for SVM [ZhangIJCV2007]

Evaluation criteria

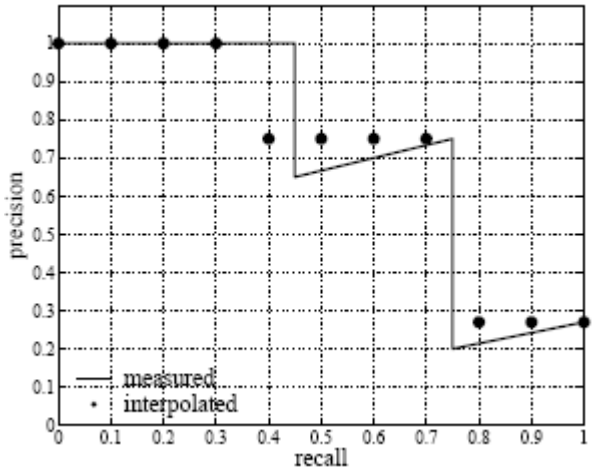
Ground truth annotation



Detection results:

- >50 % overlap of bounding box with GT
- *one* bounding box for each object
- confidence value for each detection

Precision-Recall (PR) curve:



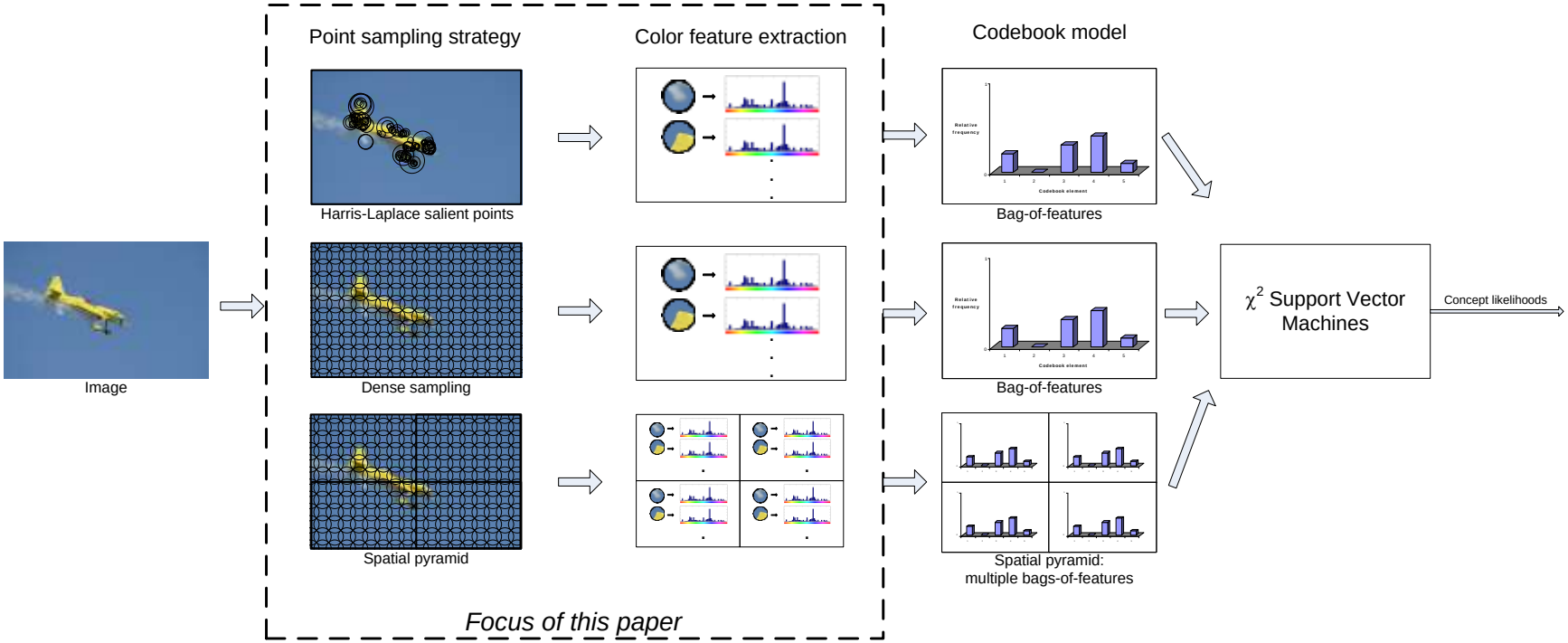
$$precision = \frac{\#corr.\,det.}{\#all\ det.}$$

$$recall = \frac{\#corr.\,det.}{\#all\ objects}$$

Average Precision (AP) value:

$$AP = \frac{1}{11} \sum p_{interp}(r),\ r \in \{0, .1, ..., 1\}$$

Pipeline: Van de Sande, Gevers: IEEE PAMI, 2010, > 2100 citations



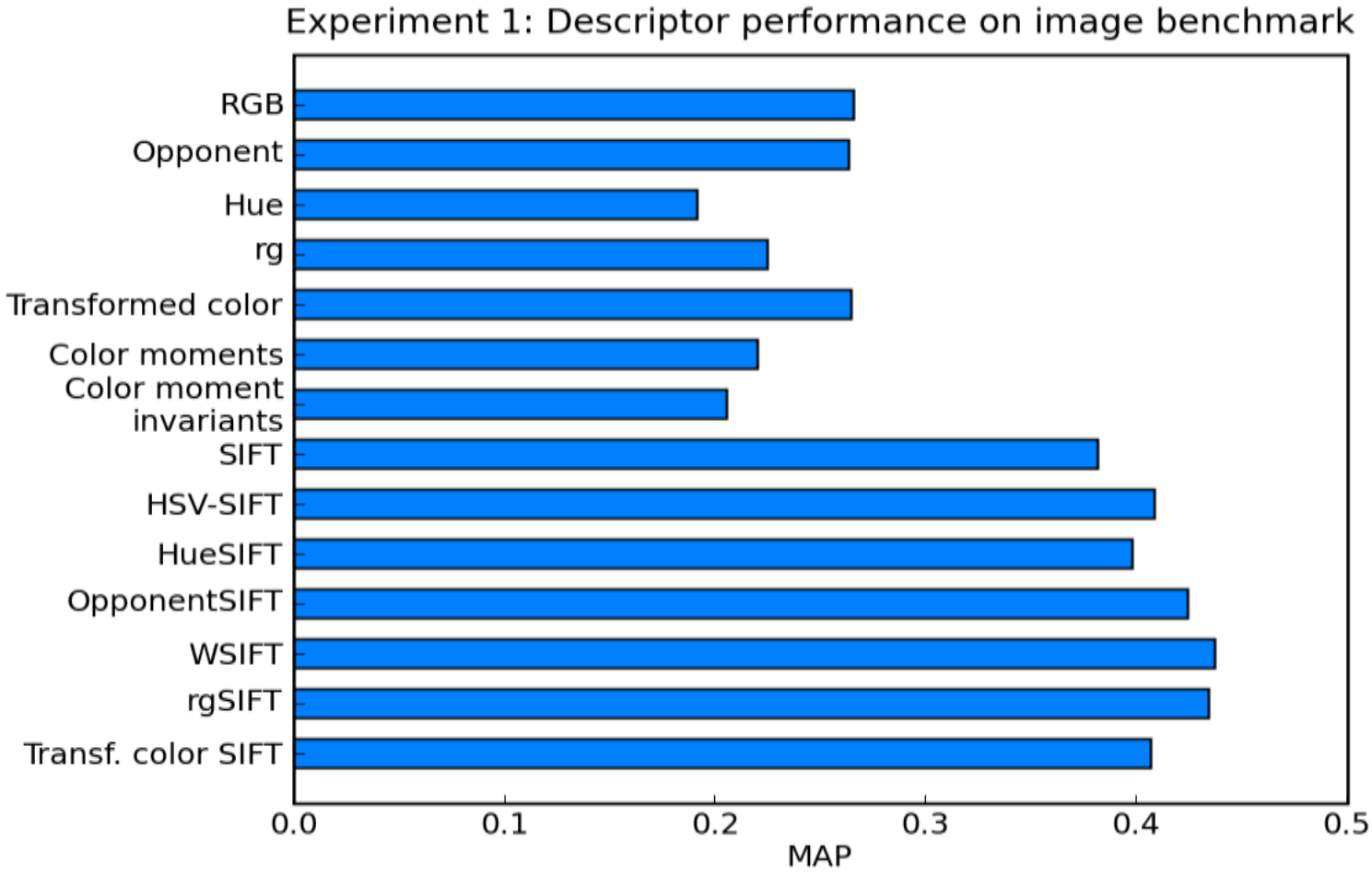
Point sampling
Harris-Laplace
Dense sampling

Spatial Pyramid
1x1
2x2
1x3

Color Descriptor
SIFT
OpponentSIFT
WSIFT
rgSIFT
Transformed color SIFT

Codebook size=4000

Results on PASCAL VOC Testset 2007

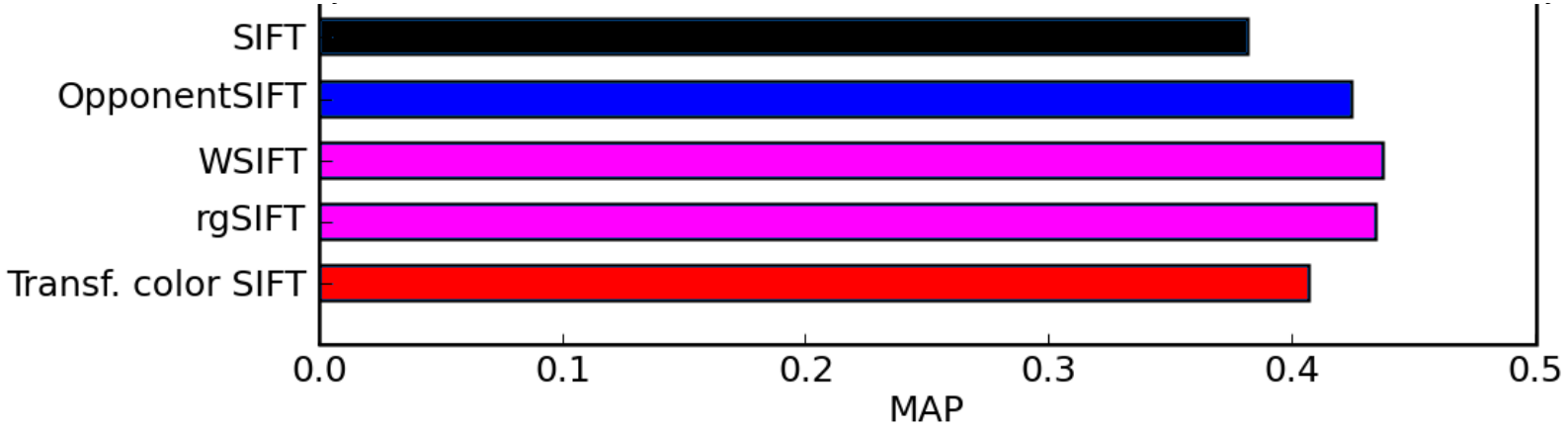


Color Descriptor Taxonomy

[van de Sande, IEEE PAMI, 09]

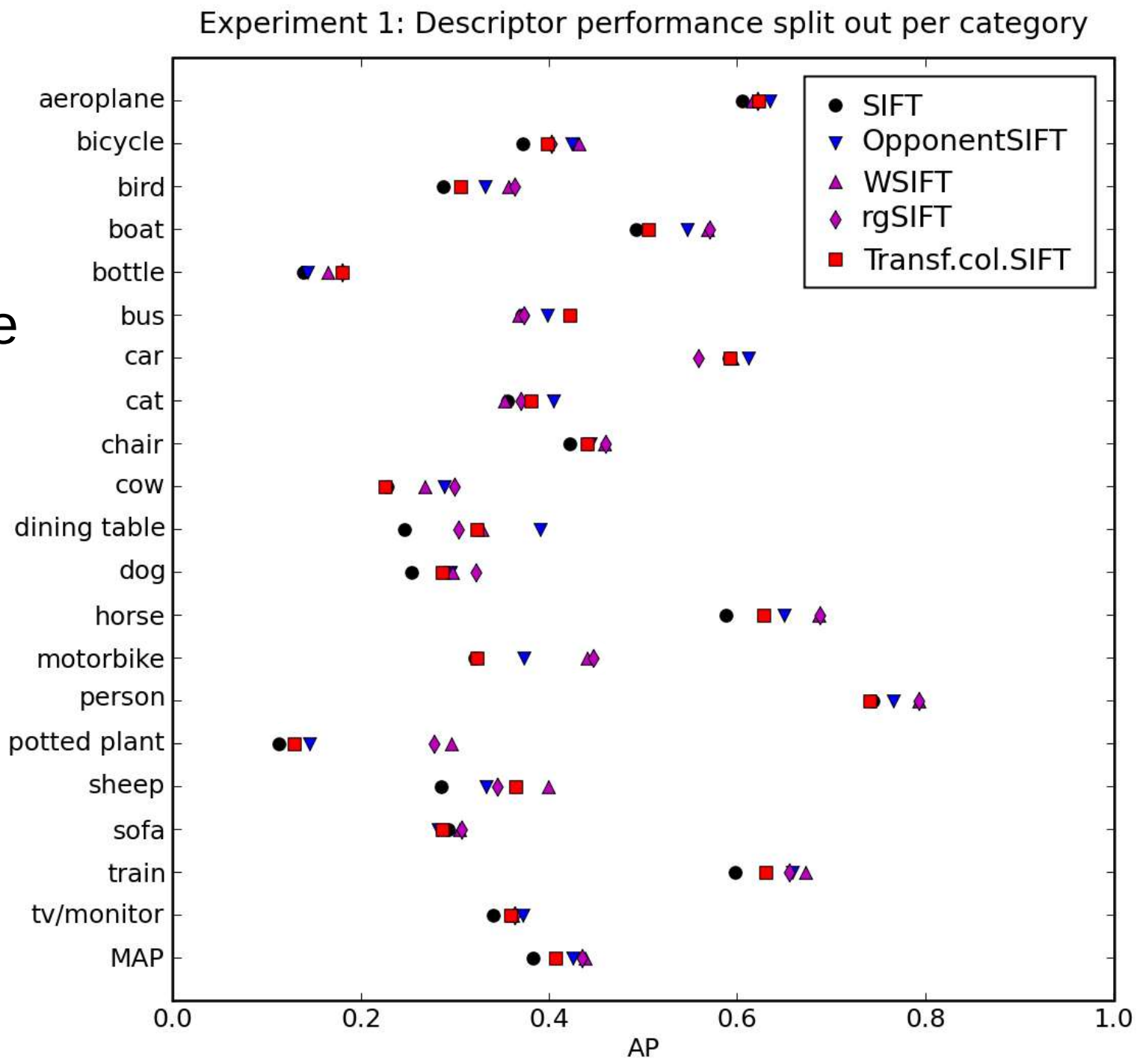
	Light intensity change $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light intensity shift $\begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light intensity change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light color change $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light color change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix}$
RGB Histogram	-	-	-	-	-
O_1, O_2	-	+	-	-	-
O_3, Intensity	-	-	-	-	-
Hue	+	+	+	-	-
Saturation	+	+	+	-	-
r, g	+	-	-	-	-
Transformed color	+	+	+	+	+
Color moments	-	+	-	-	-
Moment invariants	+	+	+	+	+
SIFT (∇I)	+	+	+	+	+
HSV-SIFT	+	+	+	+/-	+/-
HueSIFT	+	+	+	+/-	+/-
OpponentSIFT	+/-	+	+/-	+/-	+/-
W-SIFT	+	+	+	+/-	+/-
rgSIFT	+	+	+	+/-	+/-
Transf. color SIFT	+	+	+	+	+

Results on PASCAL VOC 2007



	Light intensity change $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light intensity shift $\begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light intensity change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light color change $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light color change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix}$
SIFT	+	+	+	+	-
OpponentSIFT	+	+	+	-	-
WSIFT	+	+	+	-	-
rgSIFT	+	+	+	-	-
RGB SIFT	+	+	+	+	+

Scale-invariance w.r.t. light intensity important



Discussion

- #1 model for the VOC:

Light intensity change and shift

$$\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$$

- You need scale-invariance and shift-invariance w.r.t. light intensity
- Invariance to light color is not needed and decreases the discriminative power

VOC2007 Results –Airplanes (1)

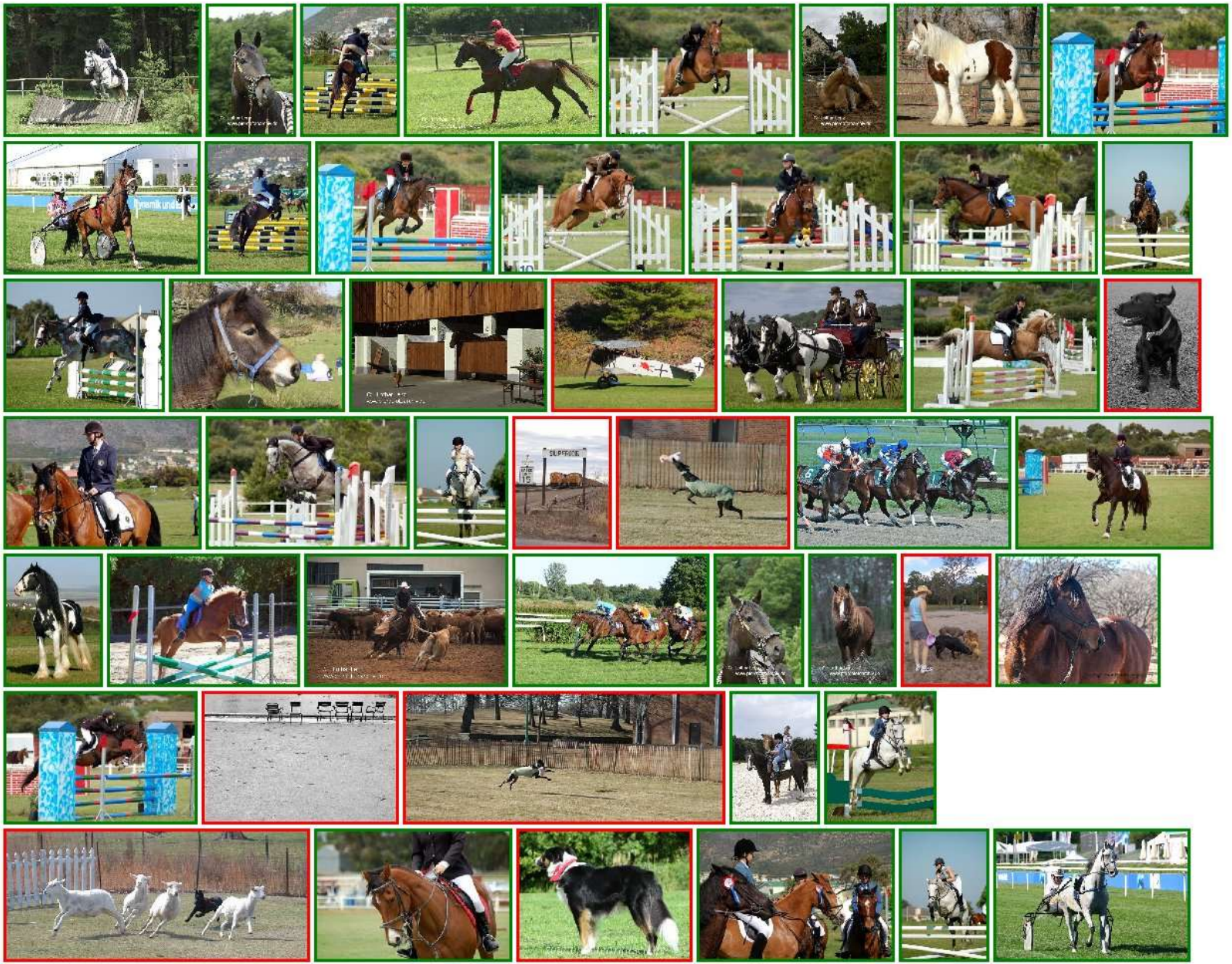
VOC2007 Results – Airplanes (2)

VOC2007 Results –Persons (1)

VOC2007 Results – Persons (2)

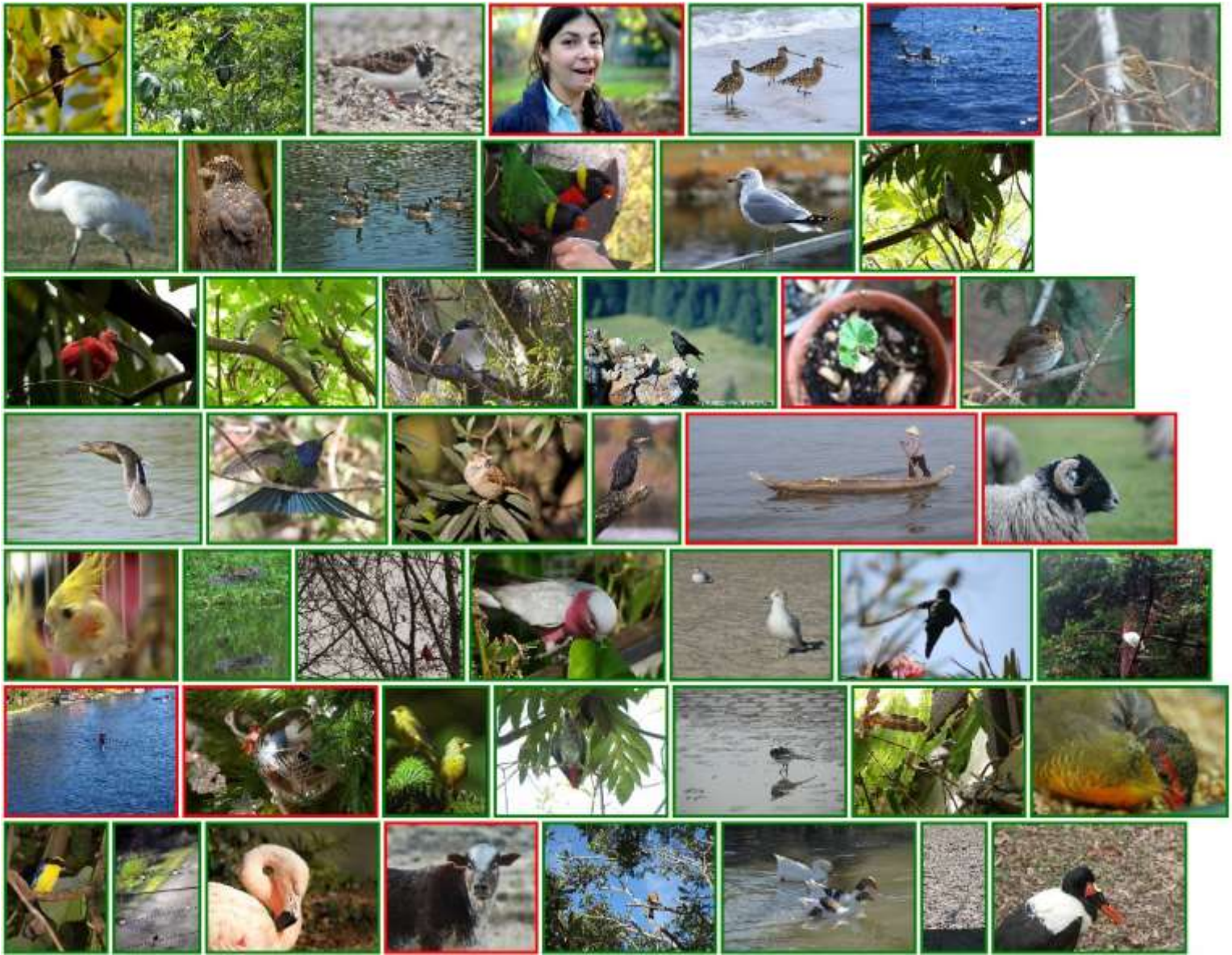
VOC2007 Results – Horses (1)

VOC2007 Results – Horses (2)

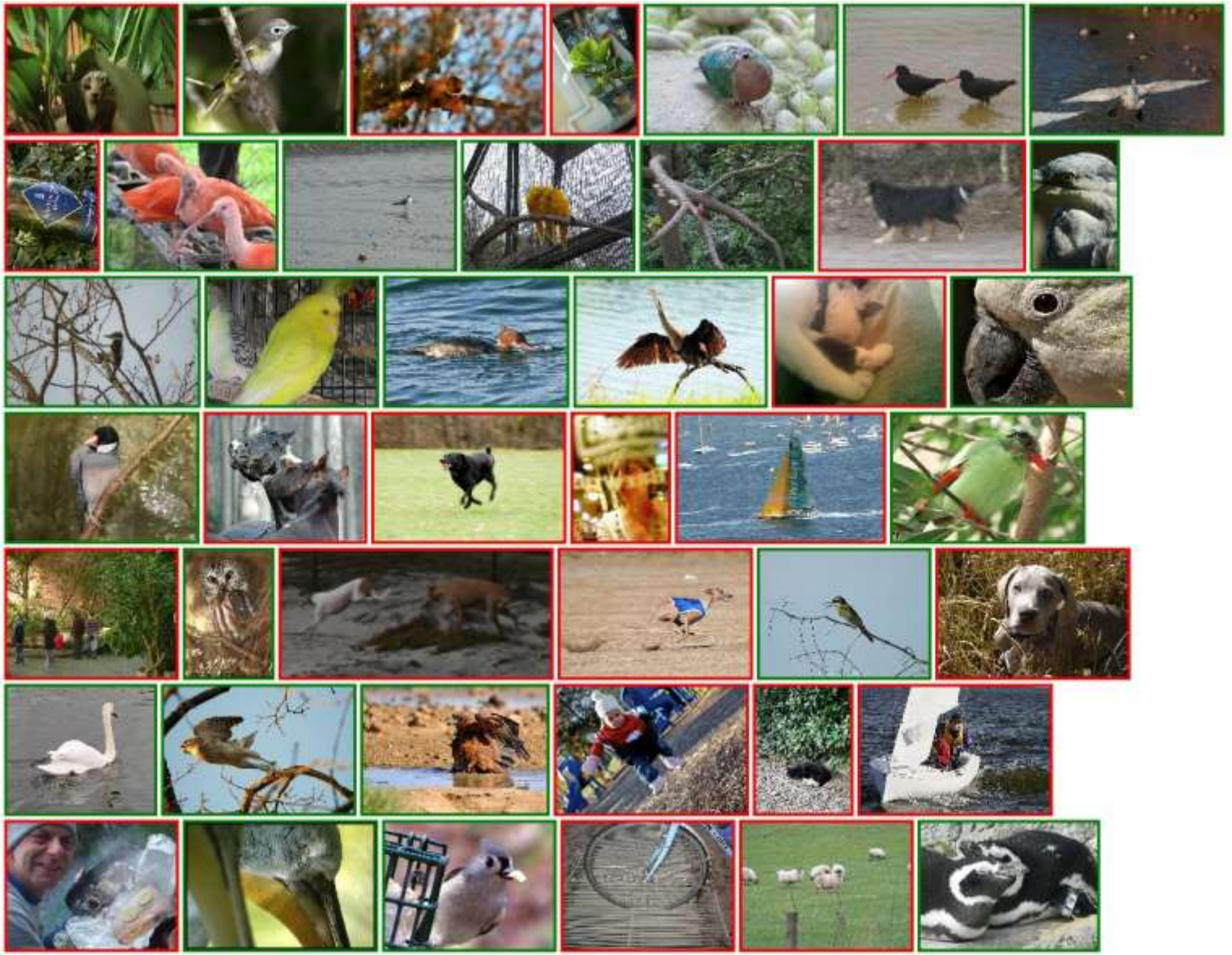


VOC2007 Results – Dogs (1)

VOC2007 Results – Birds(1)



VOC2007 Results – Birds (2)



Color Descriptors on VOC08

http://www.cat.uab.cat/Software/Pascal_Challenge/indexValidation.php

- Invariance properties of the descriptors used

	Light intensity change $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light intensity shift $\begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light intensity change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_1 \\ o_1 \end{pmatrix}$	Light color change $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$	Light color change and shift $\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} o_1 \\ o_2 \\ o_3 \end{pmatrix}$
SIFT	+	+	+	+	+
OpponentSIFT	+	+	+	-	-
WSIFT	+	+	+	-	-
*rg*SIFT	+	+	+	-	-
RGBSIFT	+	+	+	+	+

Descriptors	MAP on VOC2008val
Intensity SIFT	42,3
All five (=Soft5ColorSIFT)	45,5

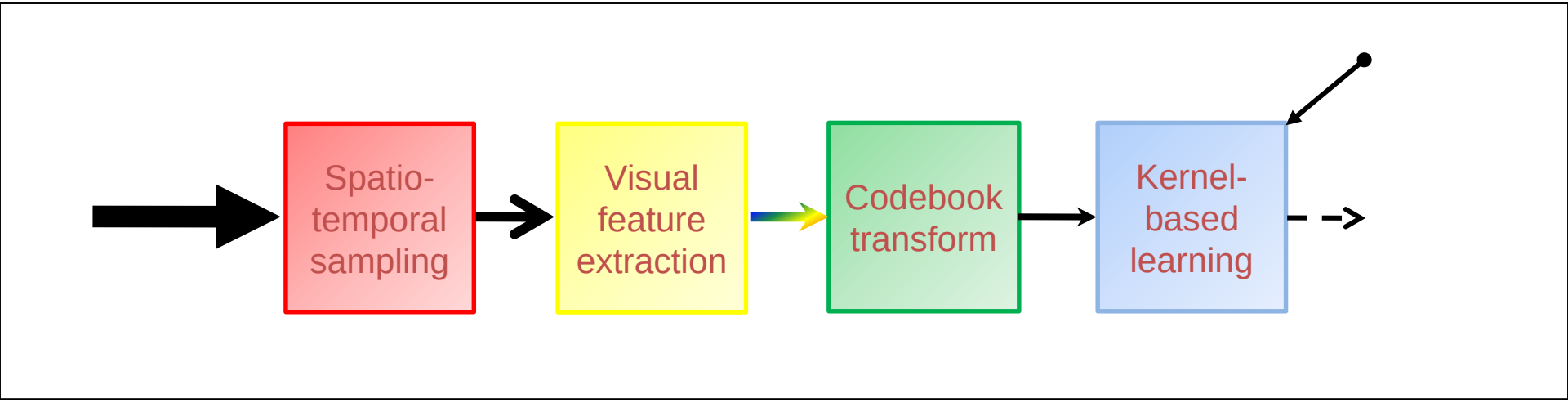
By adding color: **+8%**

TRECVid

Koen van de Sande
Cees Snoek
Jan van Gemert
Jasper Uijlings
Jan-Mark Geusebroek
Theo Gevers
Arnold Smeulders

University of Amsterdam

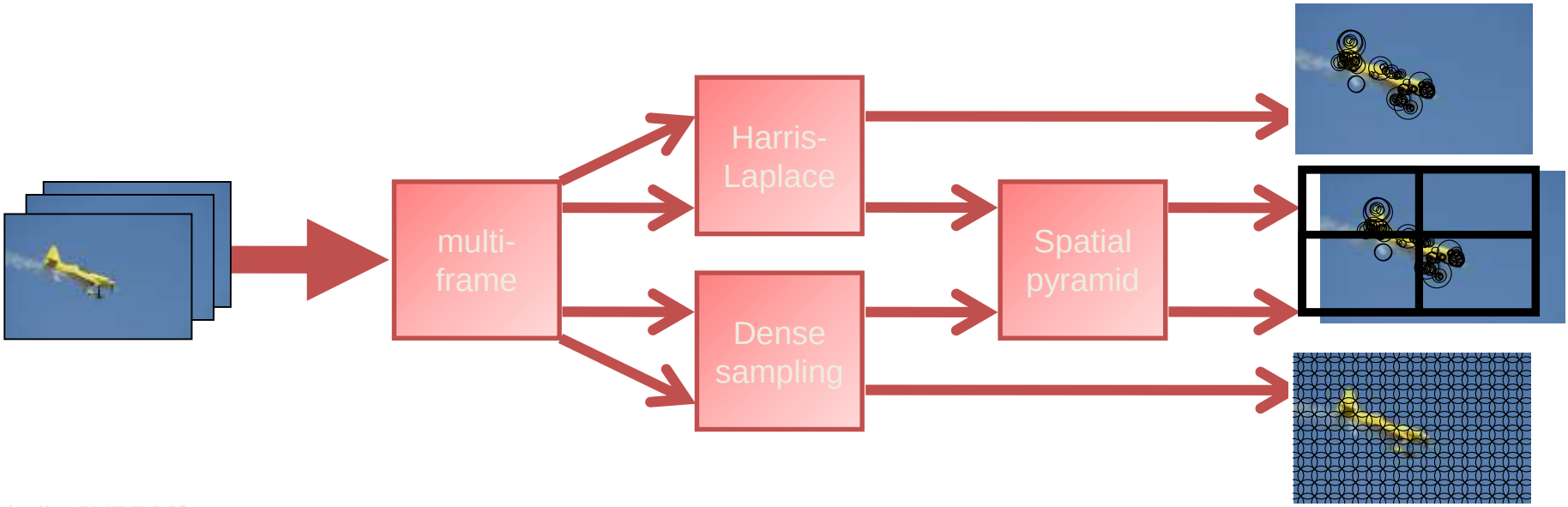
Concept Detection Stages



Spatio-Temporal Sampling

- Spatial pyramid
  - 1x1 whole image
  - 2x2 image quarters
  - 1x3 horizontal bars

- Temporal analysis of up to 5 frames per shot

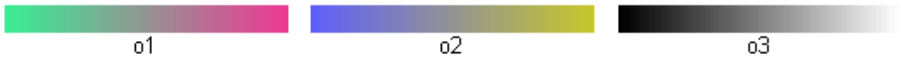


[Lazebnik, CVPR06]
[Marszalek, VOC07]

Invariant Visual Descriptors

Color SIFT:

- Intensity-based SIFT
- OpponentSIFT
- C-SIFT
- *rg*SIFT
- Transformed color SIFT



- Add color, but also keep intensity information

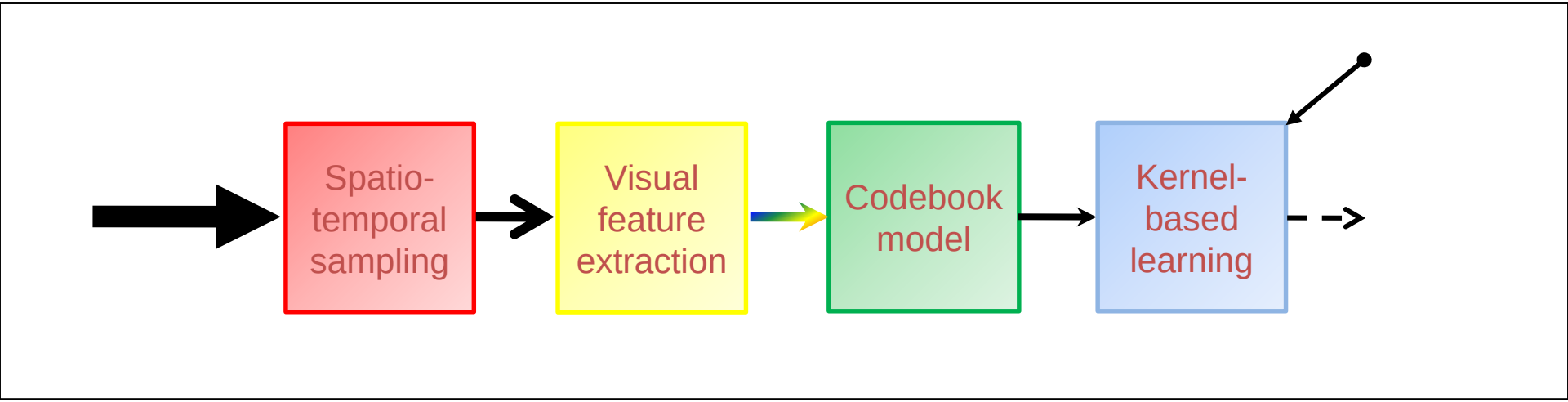
Visual Descriptors	MAP on TV2007test
Intensity SIFT	0,144
5x Color SIFT	0,155

relative +8%

TV2007test results:

- Trained on TRECVID2007 development set
- Evaluated on TRECVID2007 test set
- TRECVID2007 development + test = 2008 development

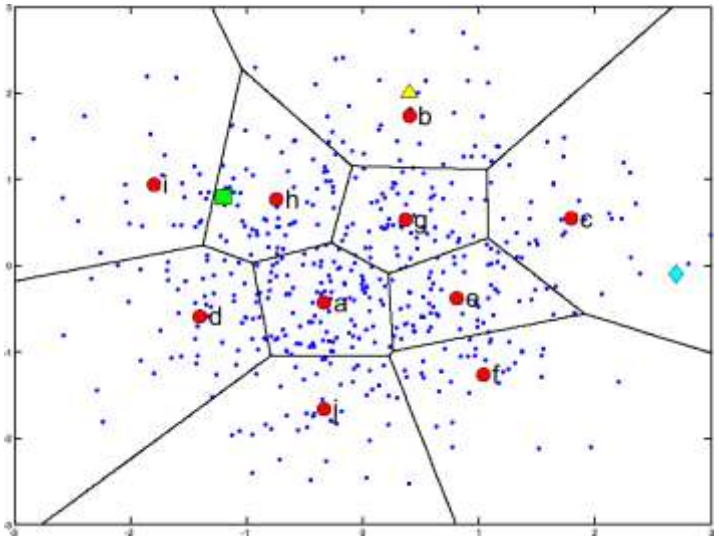
Concept Detection Stages



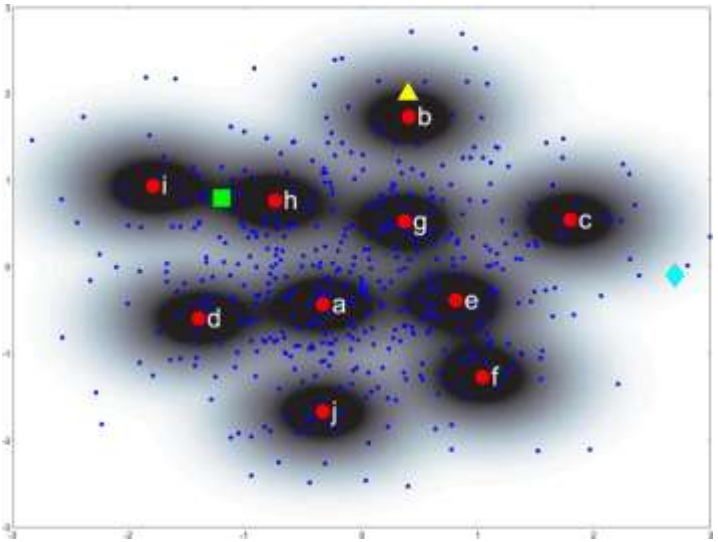
Codebook Assignment

Soft assignment using Gaussian kernel

● Codeword



Hard assignment



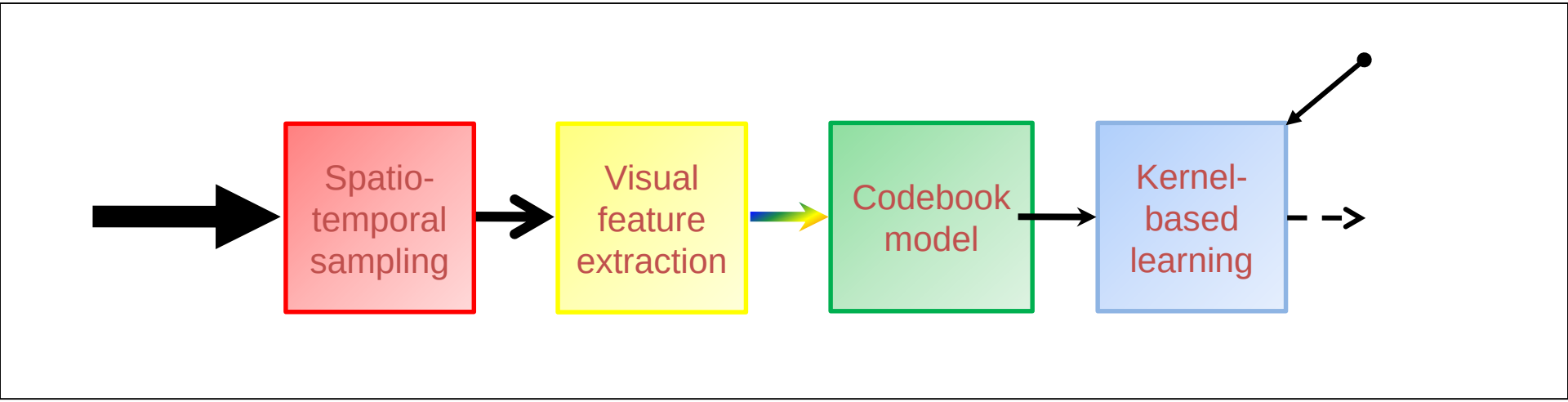
Soft assignment

Assignment	MAP on TV2007test
Hard	0,155
Soft	0,166

relative +7%

[van Gemert, ECCV08]

Concept Detection Stages



Robust Temporal Approach

- No cloud computing yet: need to be efficient ↑↑
- Process 5 frames per shot in test set
- Linear increase in computation: x5

Codebook library	Frames/shot	MiAP on TV2008test
3x Color SIFT	1	0,152
3x Color SIFT	5	0,184

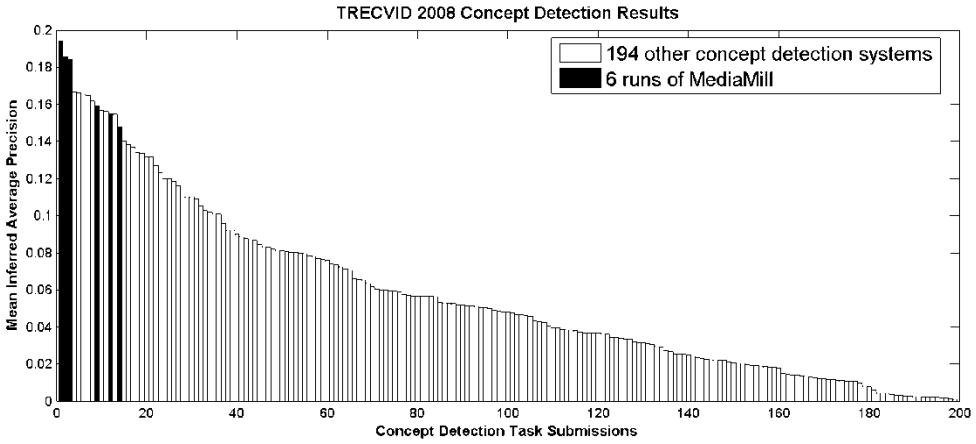
relative +20%

- In 2005 paper 7.5% to 38% improvement noted for multi-frame (worst-case vs. best-case using oracle)
- **Robust color SIFT *with* temporal = ~20% improvement**

[Snoek, ICME05]

Conclusion (1)

- Illumination conditions affect concept detection
- SIFT+colorSIFT improves ~8%
- Soft codebook assignment improves ~7%
- Robust colorSIFT with simple multi-frame improves ~20%:
  - Room for more advanced methods in TRECVID 2009
- Precomputed kernel matrix reduces SVM computation time
- Near-duplicates from trailers hamper progress:
  - We suggest to exclude them, or count only once

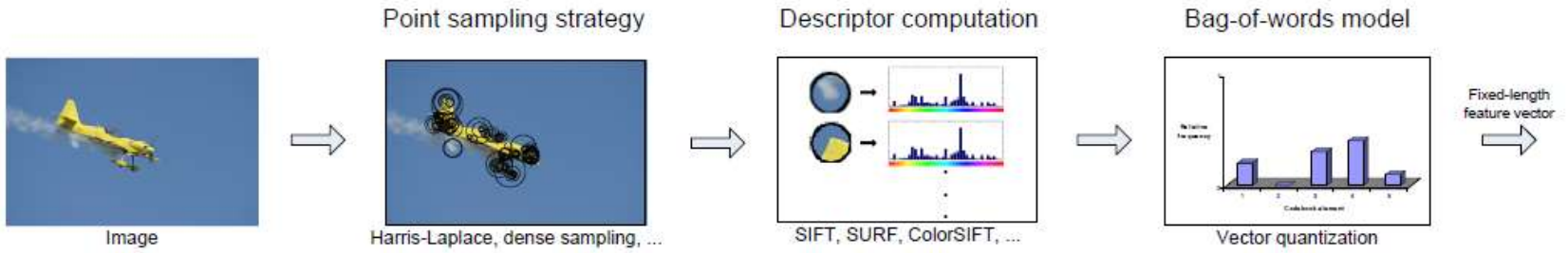


Conclusion (2)

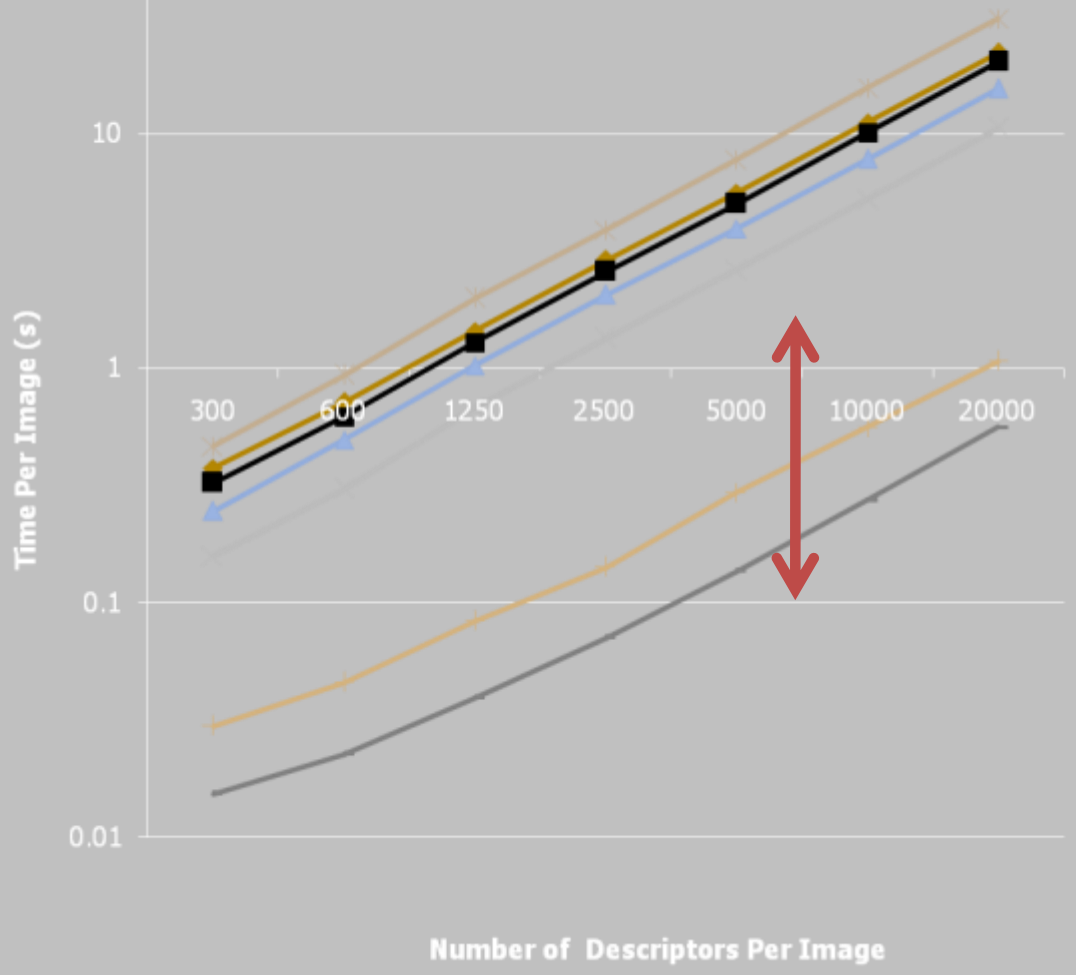
- Bag-of-word approach works
- Good local descriptors: SIFT, OpponentSIFT, rgSIFT/WSIFT, RGB SIFT
- Combining these color features gives state-of-the-art performance: 1ste position in VOC08 and second in VOC09.
- Drawback: computational costs of bag-of-word approach

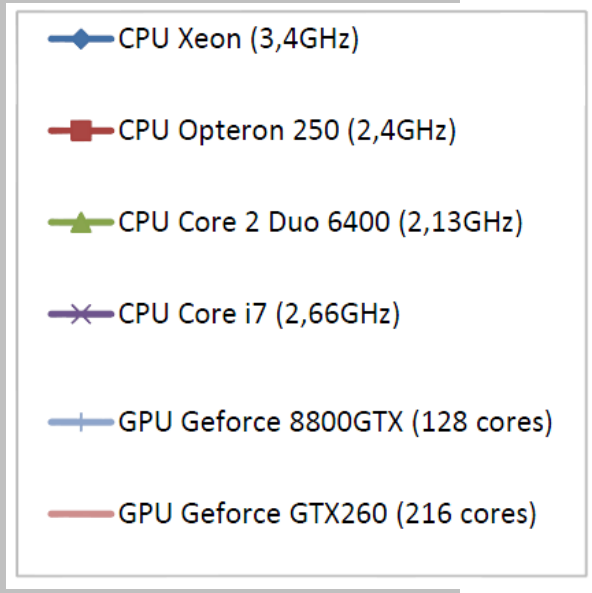
GPU-Accelerated Feature Extraction

- Single bag-of-words feature up to 15s/frame (CPU-time)
- TRECVid 2008 / PASCAL VOC 2008 consortium entries used 10 of these features
- More than 80% of time spent in vector quantization



Vector Quantization Timings for ColorSIFT

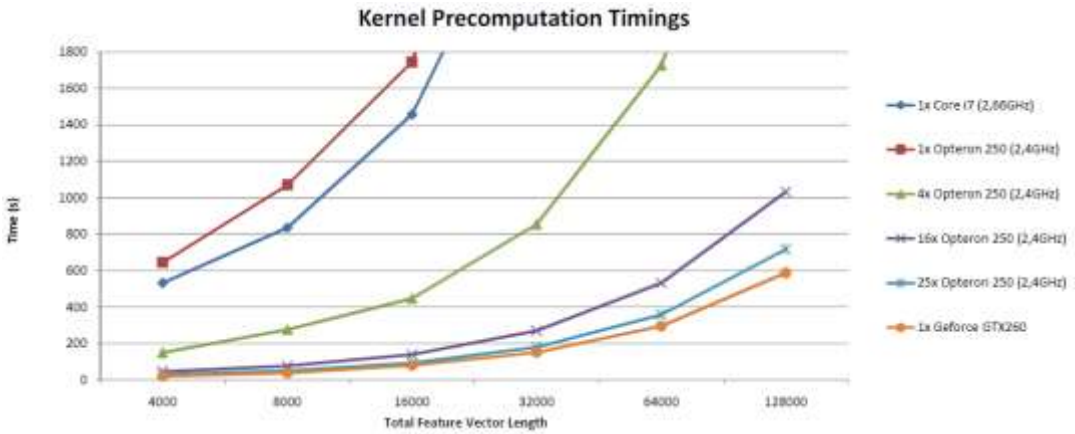




17x speedup
CPU vs GPU

Kernel Value Precomputation

- Step from image feature vectors to kernel-based classifiers from WP5 (SVM/SR-KDA)
- Computes χ^2 distance between pairs of images
- Suitable for GPU implementation: **22x speedup**
- TRECVid 2008 processing time: 800 CPU hours vs. 37 GPU hours



Process datasets order of magnitude larger

or

Single GPU replaces medium-sized cluster

Conclusions

- Large scale datasets with annotations
- Color and photometric invariance needed
- Balance between discriminative power and invariance
- Color add information to classification achieving best performance in VOC08/VOC09, TRECVid08/TRECVid09 and ImageCLEF.
- Speed up is required (e.g. GPU)

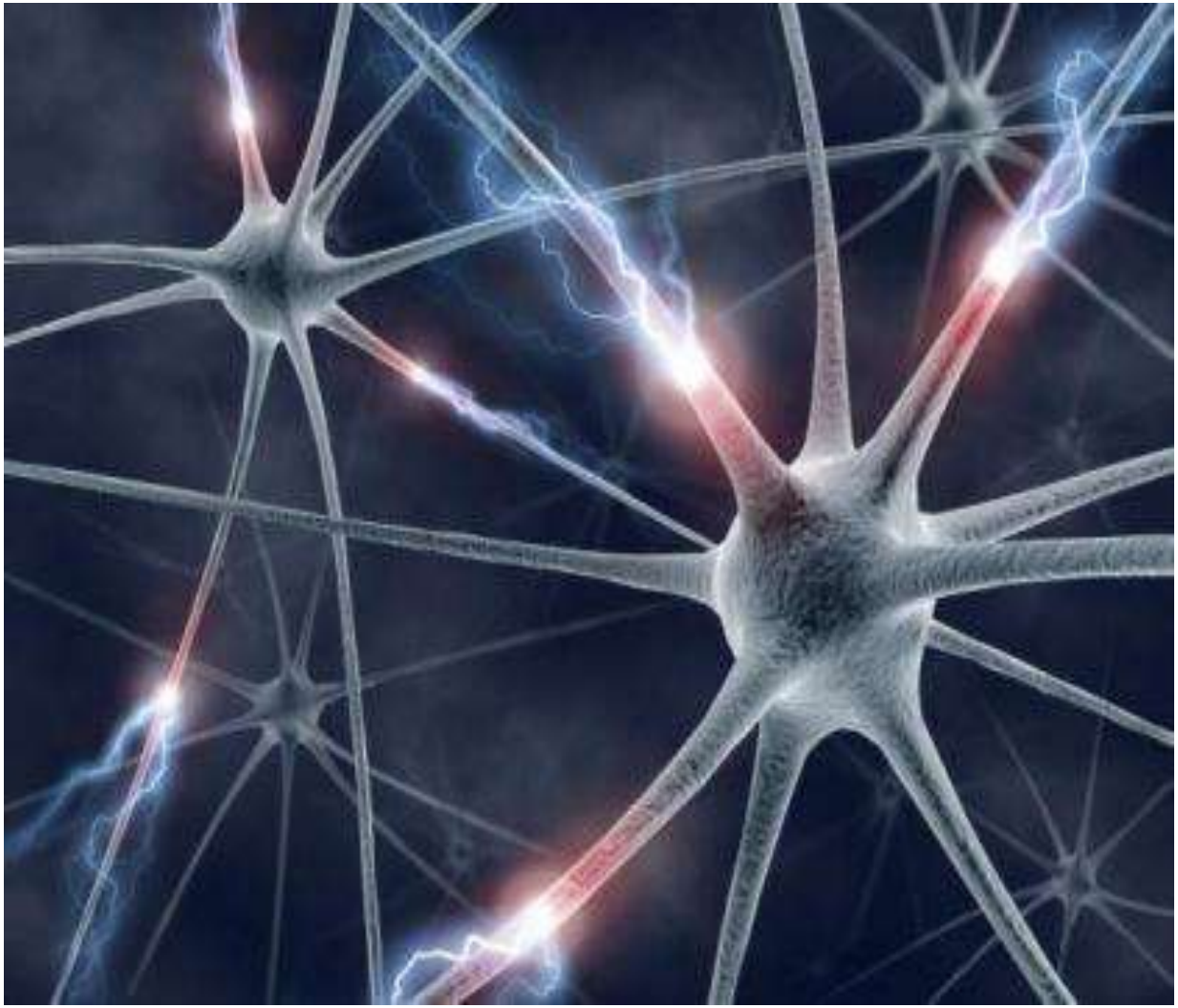
Today's class

Segmentation

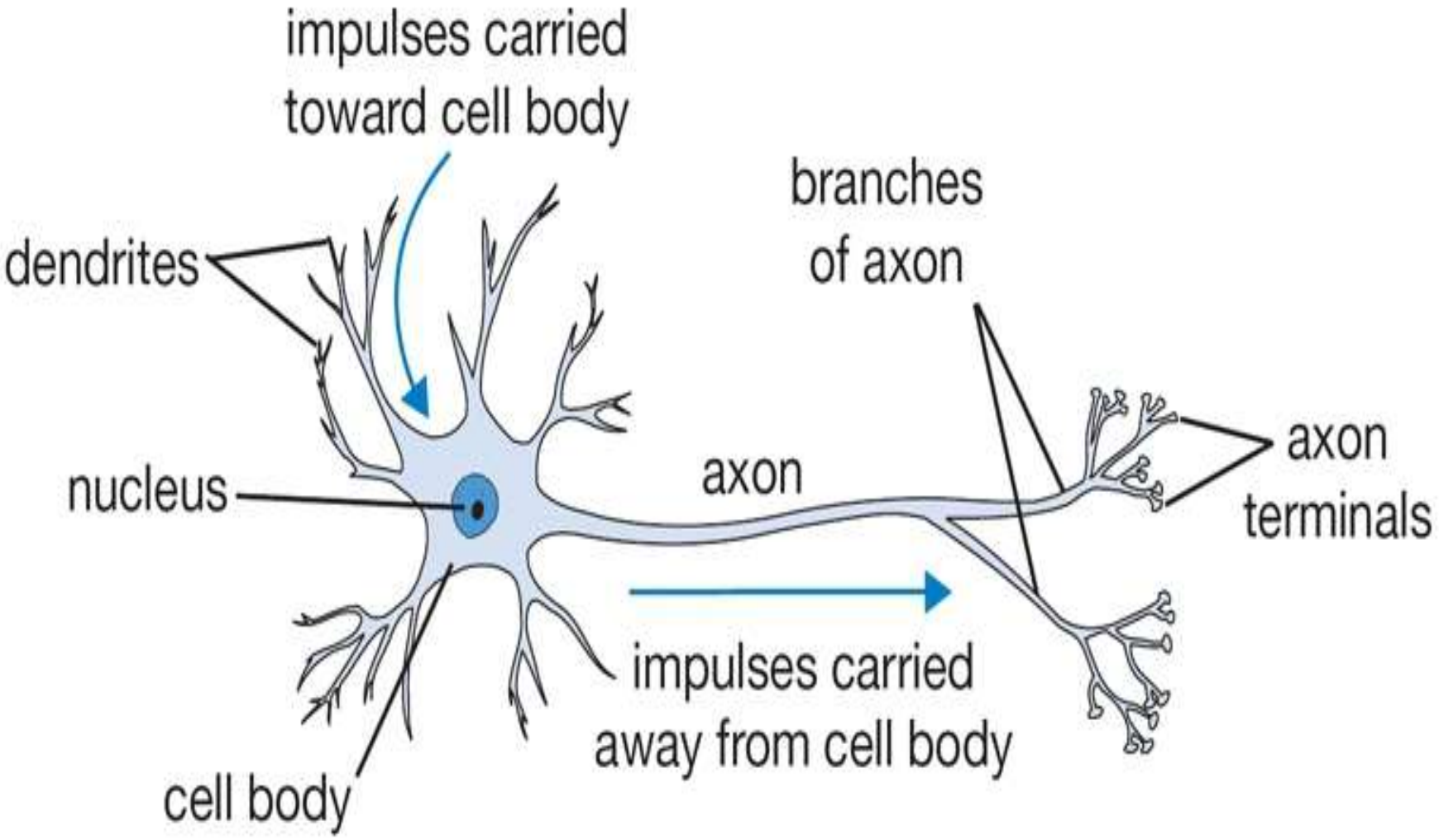
Object Recognition

Bag-of-Words

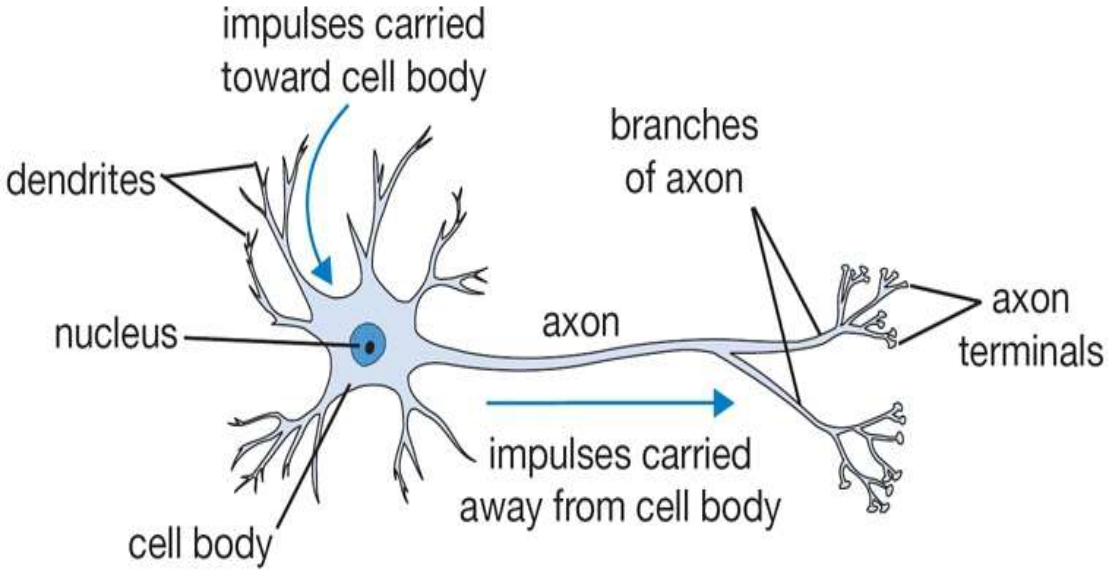
Deep Learning

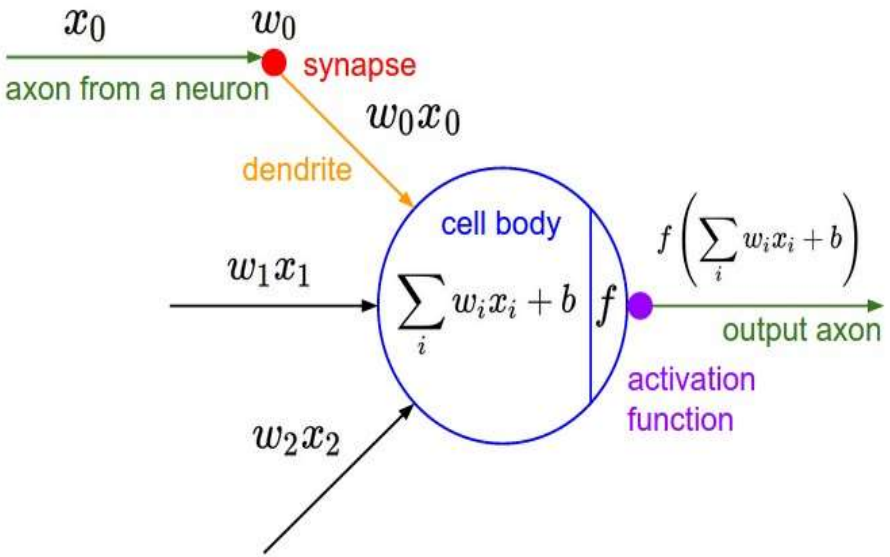


The Neuron


impulses carried
toward cell body
dendrites
branches
of axon
nucleus
axon
axon
terminals
impulses carried
away from cell body
cell body

Artifical Model (1943 McCulloch/Pitts)





The Neuron

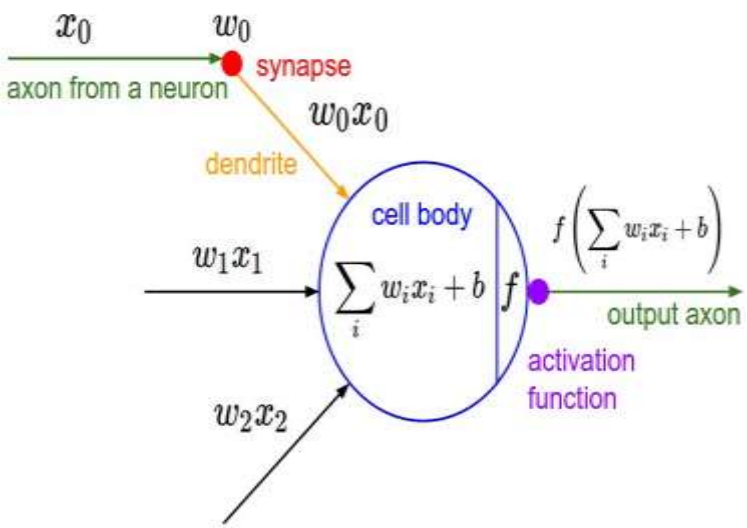
Inputs: $\vec{x} = x_1, x_1, \ldots, x_n$

Weights: $\vec{w} = w_1, w_1, \ldots, w_n$

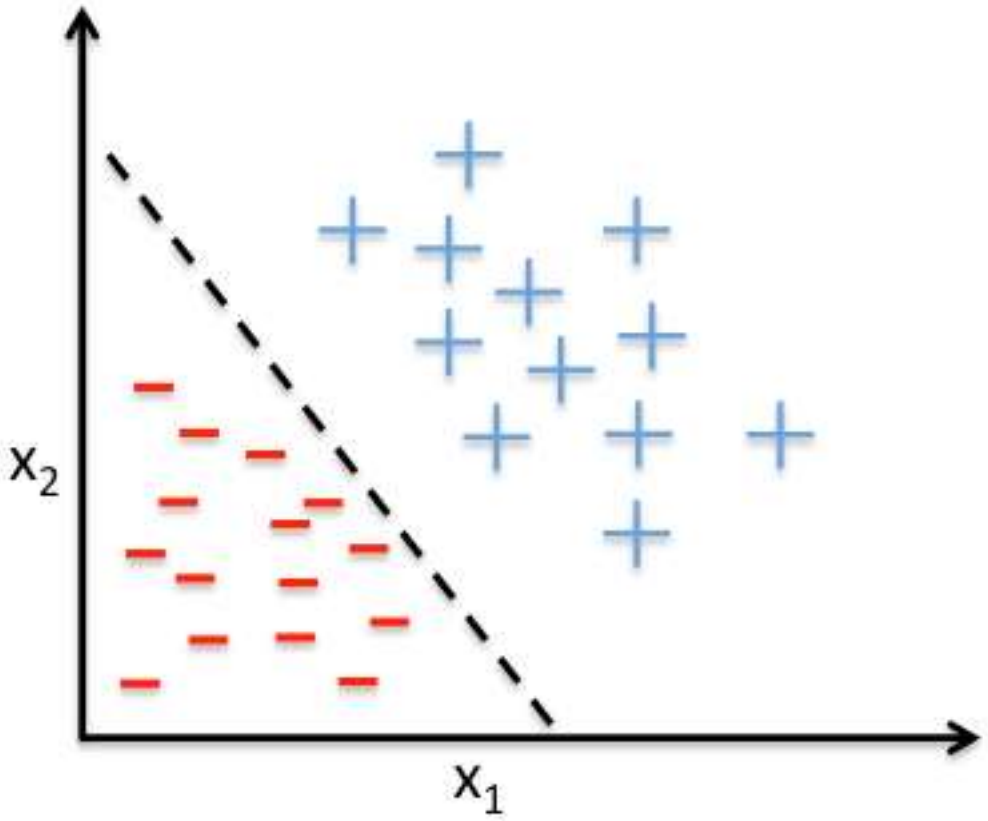
Logit: $z = \sum_{i=1}^{n} w_i x_i + b$

Output: $\mathrm{y} = f(z)$

Output: $\mathrm{y} = f(\vec{x} \cdot \vec{w} + b)$



Linear Perceptron



Example of a linear decision boundary for binary classification.

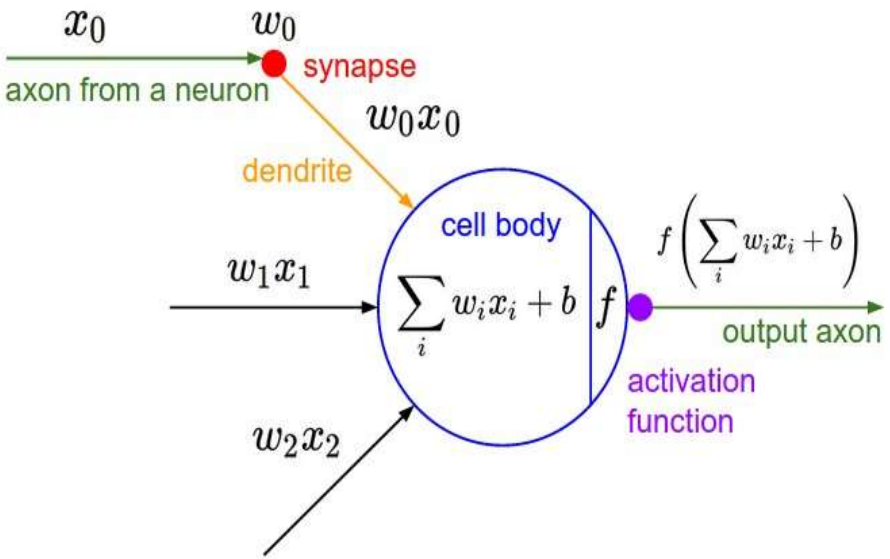
Inputs $x_1, x_1, \ldots, x_n$

Weights $w_1, w_1, \ldots, w_n$

Logit $z = \sum_{i=1}^{n} w_i x_i + b$

$$f(z) = \begin{cases} -1 & \text{if } z < 0 \\ 1 & \text{if } z \geq 0 \end{cases}$$

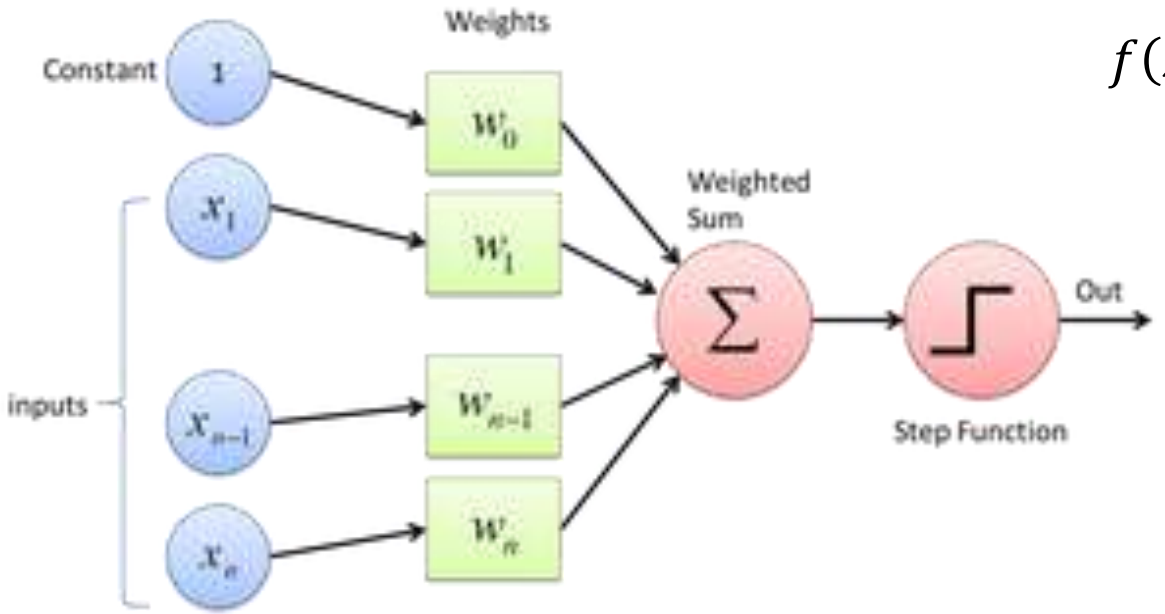
Linear Perceptron



Inputs $x_1, x_1, \ldots, x_n$

Weights $w_1, w_1, \ldots, w_n$

Logit $z = \sum_{i=1}^{n} w_i x_i + b$



$$f(z) = \begin{cases} -1 & \text{if } z < 0 \\ 1 & \text{if } z \geq 0 \end{cases}$$

Linear Neurons

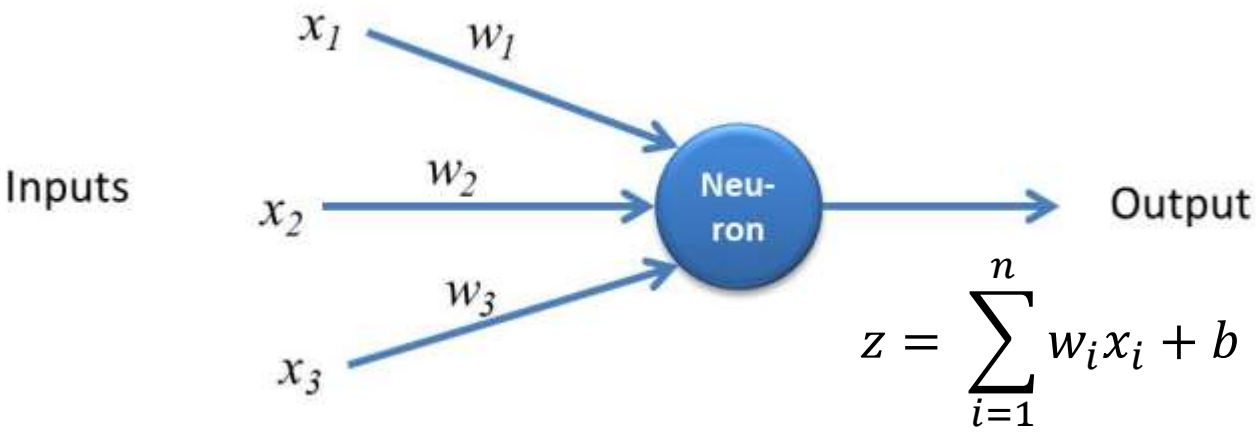
$$y = f(z) = z$$

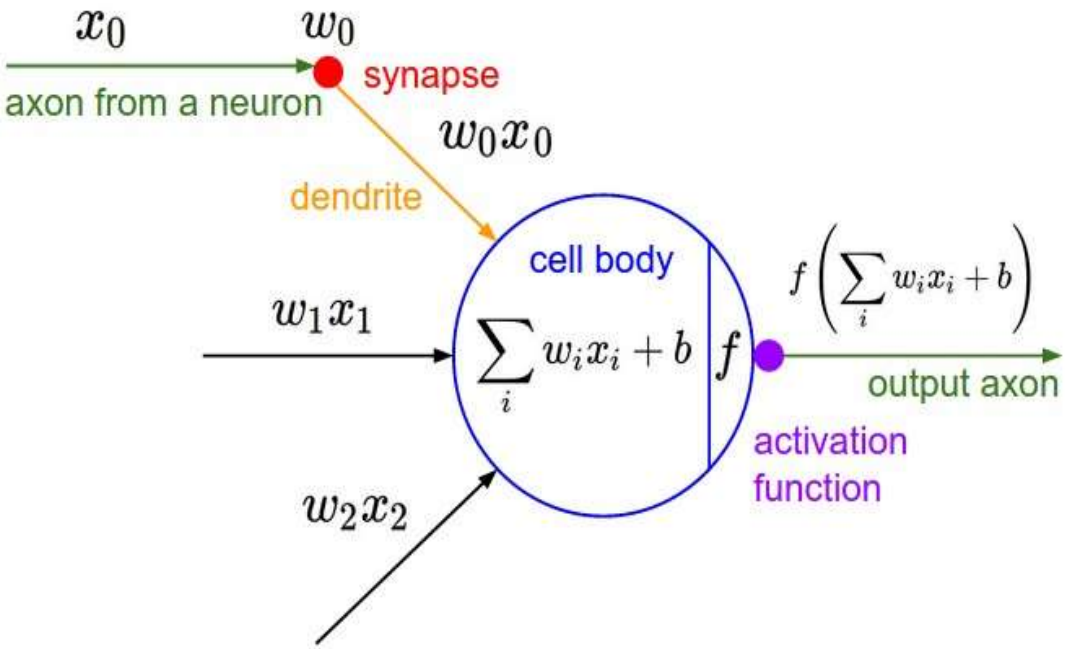
$$y = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} [w_1, w_2, \ldots w_n]$$

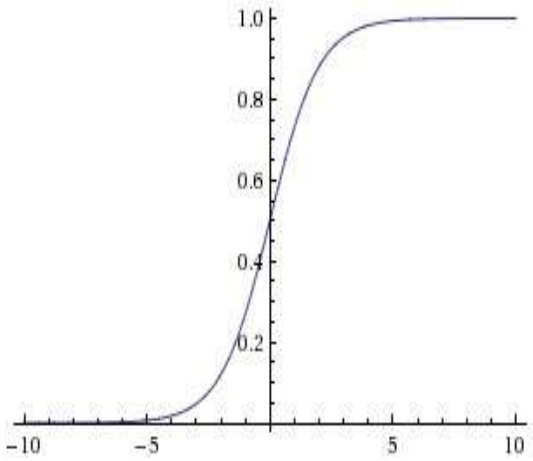
Inputs $x_1, x_1, \ldots, x_n$

Weights $w_1, w_1, \ldots, w_n$

Logit $z = \sum_{i=1}^{n} w_i x_i + b$





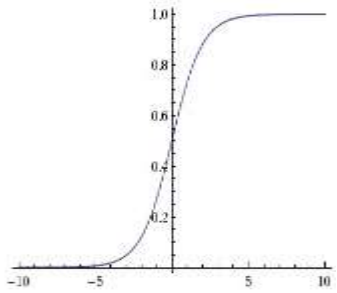


sigmoid activation function

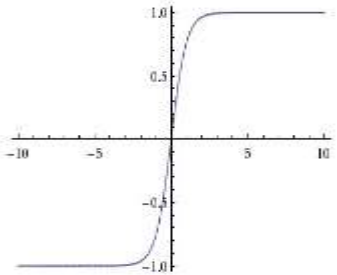
$$\frac{1}{1+e^{-x}}$$

Sigmoid

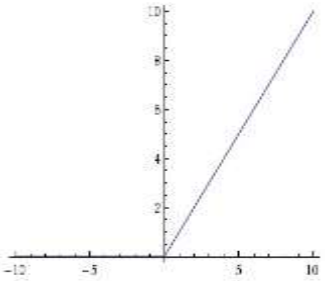
$$\sigma(x) = 1/(1+e^{-x})$$



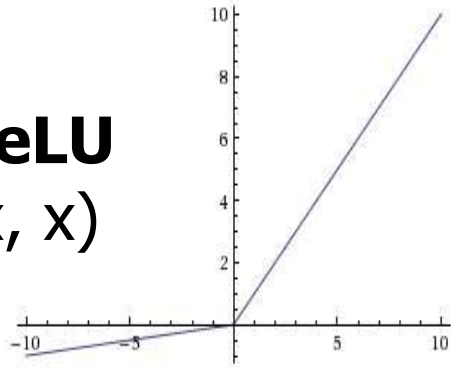
tanh tanh(x)



ReLU max(0,x)

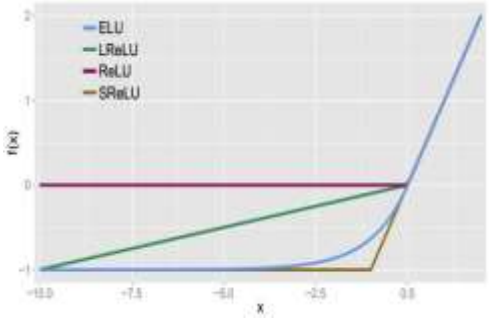


Leaky ReLU
max(0.1x, x)



Maxout $\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU $f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha\,(\exp(x) - 1) & \text{if } x \le 0 \end{cases}$



Feed-forward Neural Networks

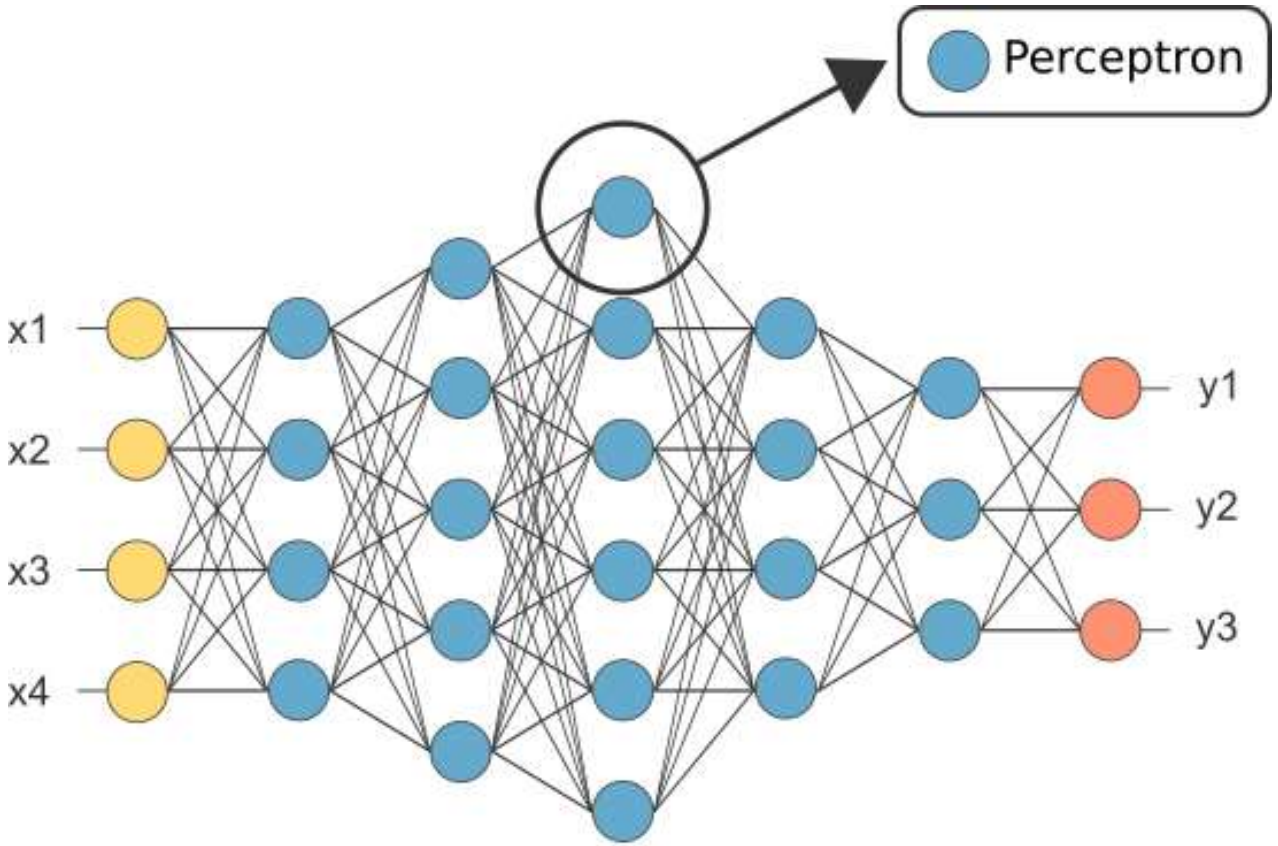
$$\vec{y} = f(\overrightarrow{W} \cdot \vec{x} + \vec{b})$$

Weight matrix nxm: $\overrightarrow{W}$

Inputs: $\vec{x} = x_1, x_1, \ldots, x_n$

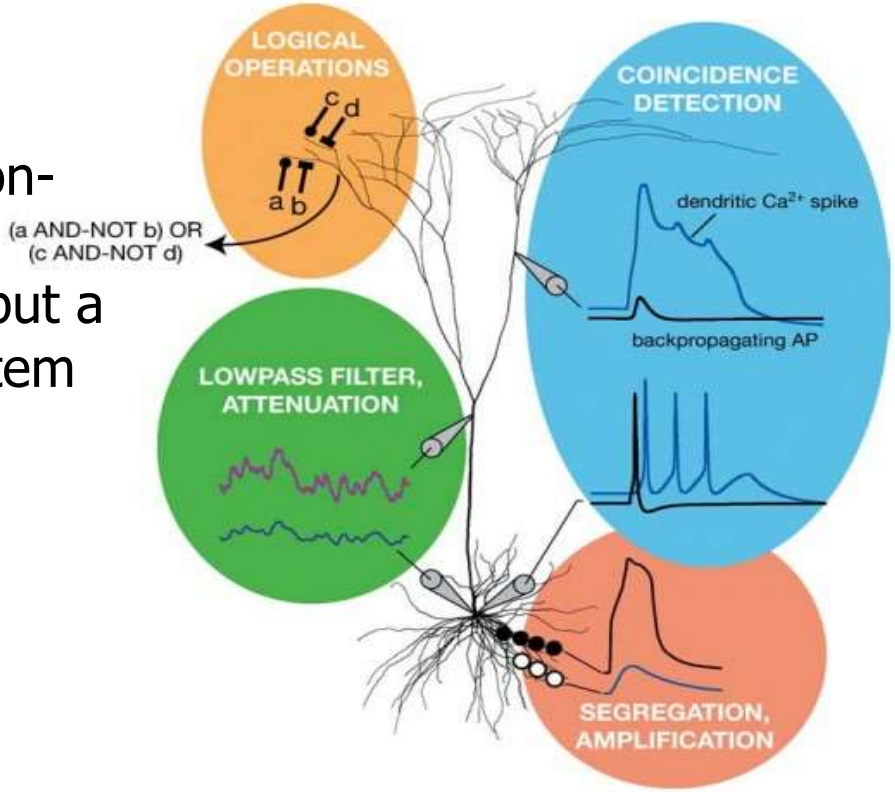
Ouputs: $\vec{y} = y_1, y_1, \ldots, y_m$

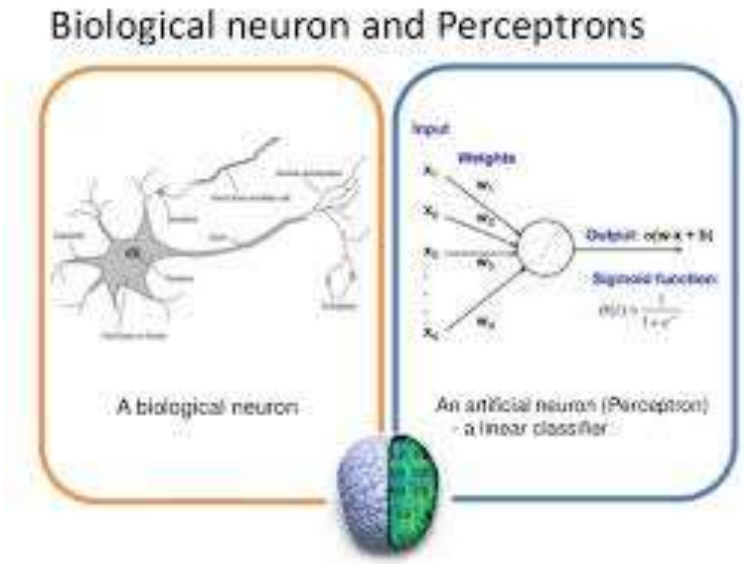
Bias: $\vec{b}$



Biological Neurons:

- Many different types
- Dendrites can perform complex non-linear computations
- Synapses are not a single weight but a complex non-linear dynamical system
- Rate code may not be adequate





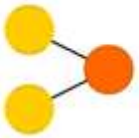
[Dendritic Computation. London and Hausser]

A mostly complete chart of

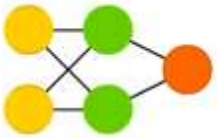
Neural Networks

- Backfed Input Cell
- Input Cell
- Noisy Input Cell
- Hidden Cell
- Probablistic Hidden Cell
- Spiking Hidden Cell
- Output Cell
- Match Input Output Cell
- Recurrent Cell
- Memory Cell
- Different Memory Cell
- Kernel
- Convolution or Pool

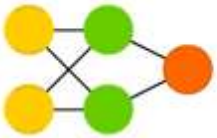
Perceptron (P)



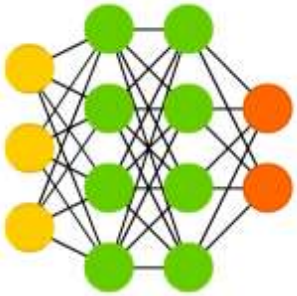
Feed Forward (FF)



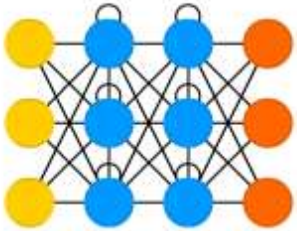
Radial Basis Network (RBF)



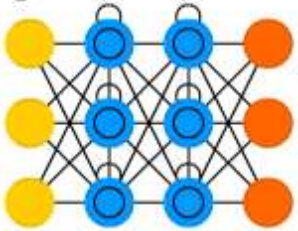
Deep Feed Forward (DFF)



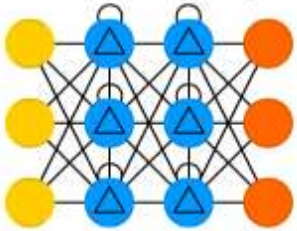
Recurrent Neural Network (RNN)



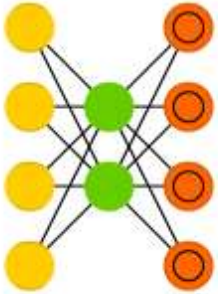
Long / Short Term Memory (LSTM)



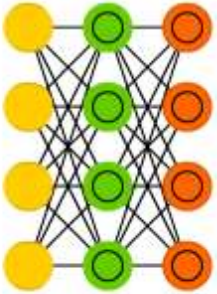
Gated Recurrent Unit (GRU)



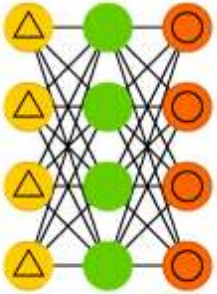
Auto Encoder (AE)



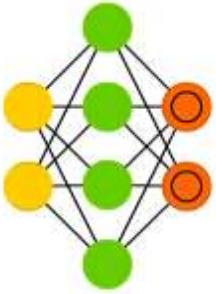
Variational AE (VAE)



Denoising AE (DAE)



Sparse AE (SAE)



Single layer feed-forward NN

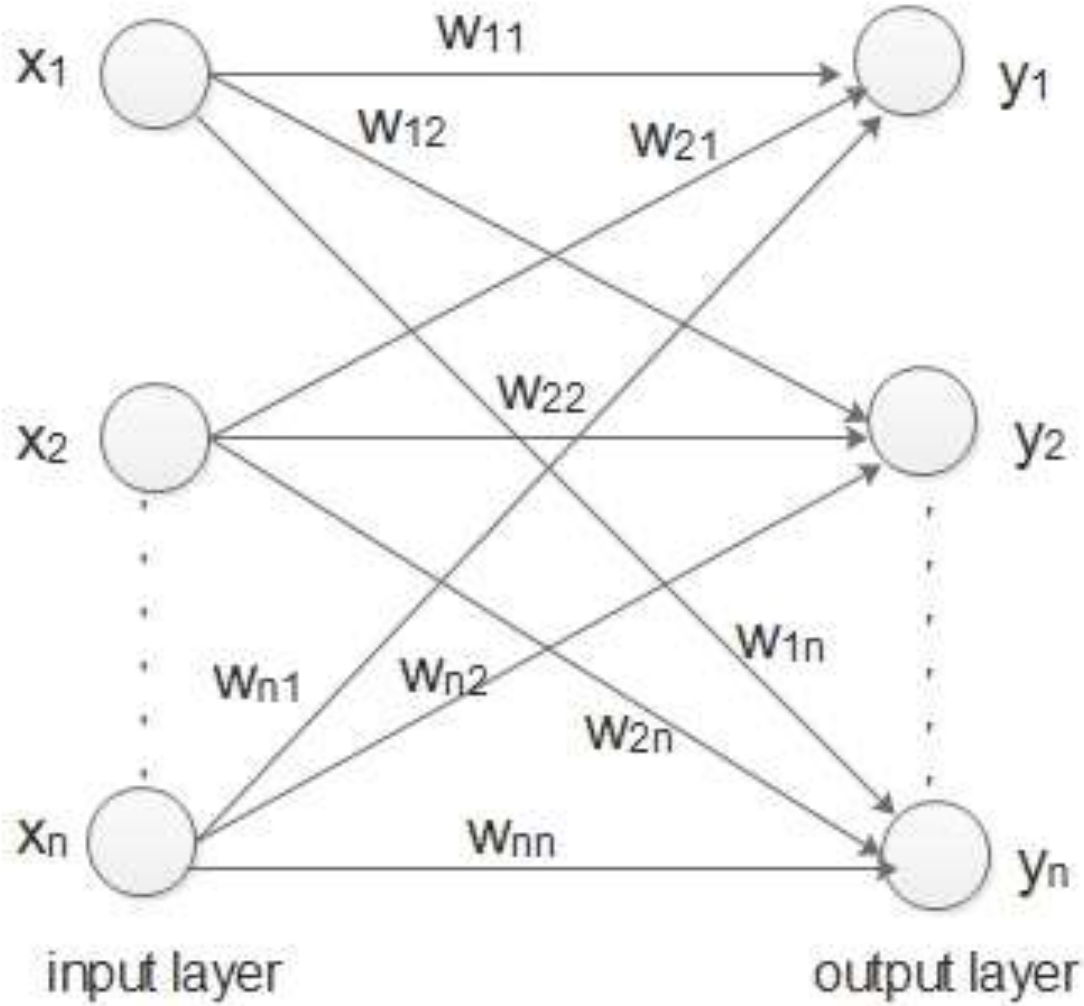


Figure: Single layer feed forward NN

CIFAR-10

10 labels
50,000 training images, each image is tiny: 32x32
10,000 test images.

$$\vec{y} = f(\vec{W} \cdot \vec{x} + \vec{b})$$

10 numbers, indicating class scores

[32x32x3]
array of numbers 0...1
(3072 numbers total)

Output=10x1 **Weights=10x3072** **Bias=10x1**

$$\vec{y} = f(\vec{W} \cdot \vec{x} + \vec{b})$$

Input=3072x1

10 numbers, indicating class scores

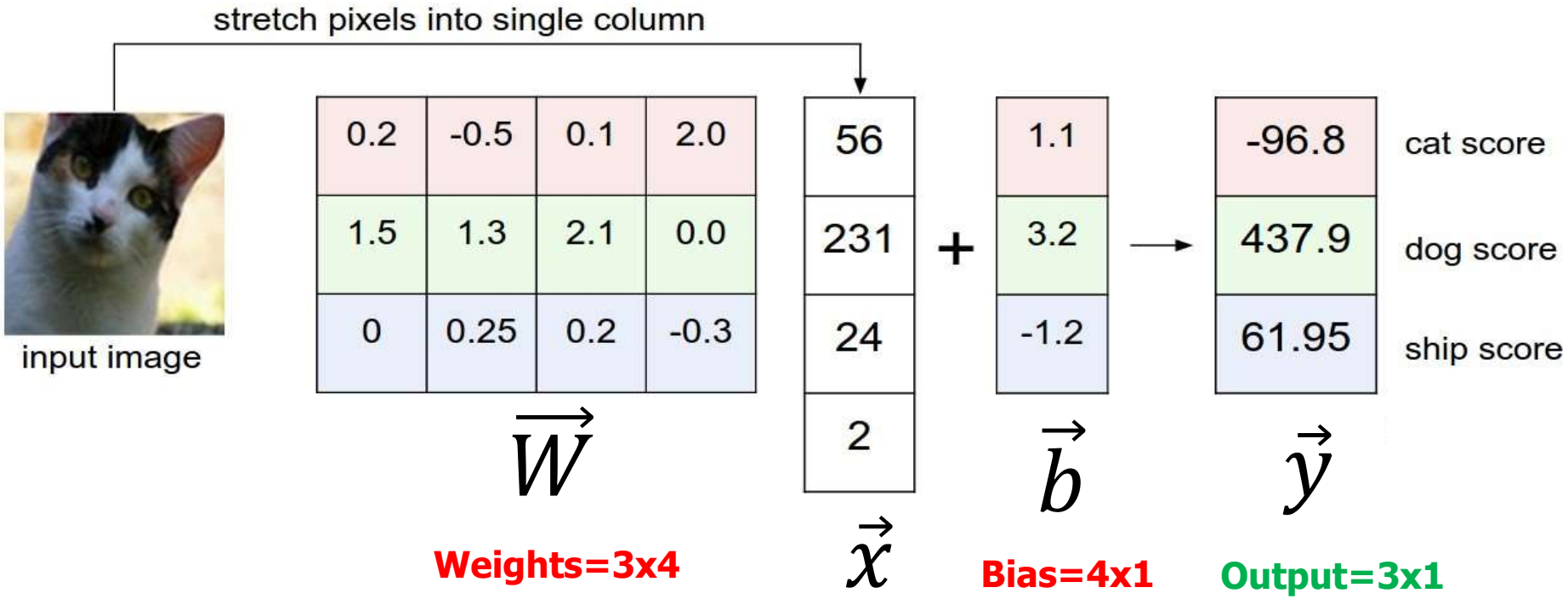
[32x32x3]
array of numbers 0...1
(3072 numbers total)

Output=3x1 Weights=3x4 Bias=4x1

$$\vec{y} = f(\vec{W} \cdot \vec{x} + \vec{b})$$

Input=4x1

(cat/dog/ship)

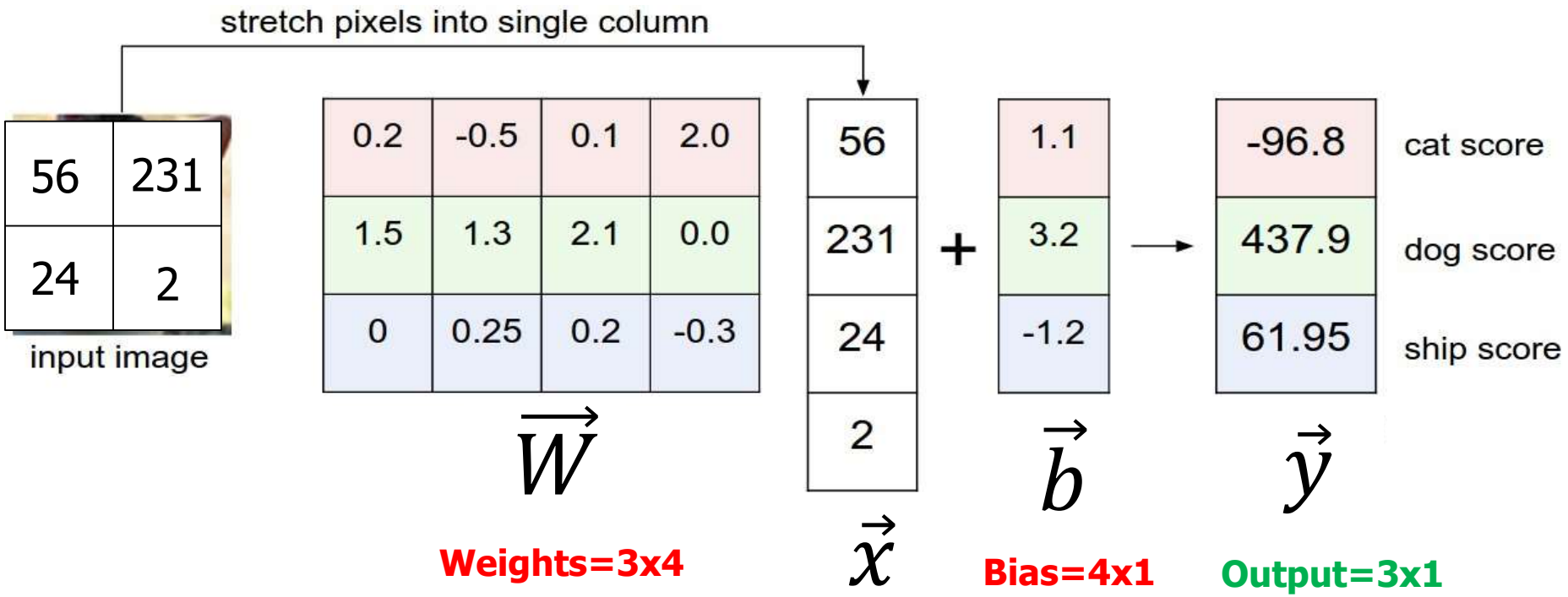


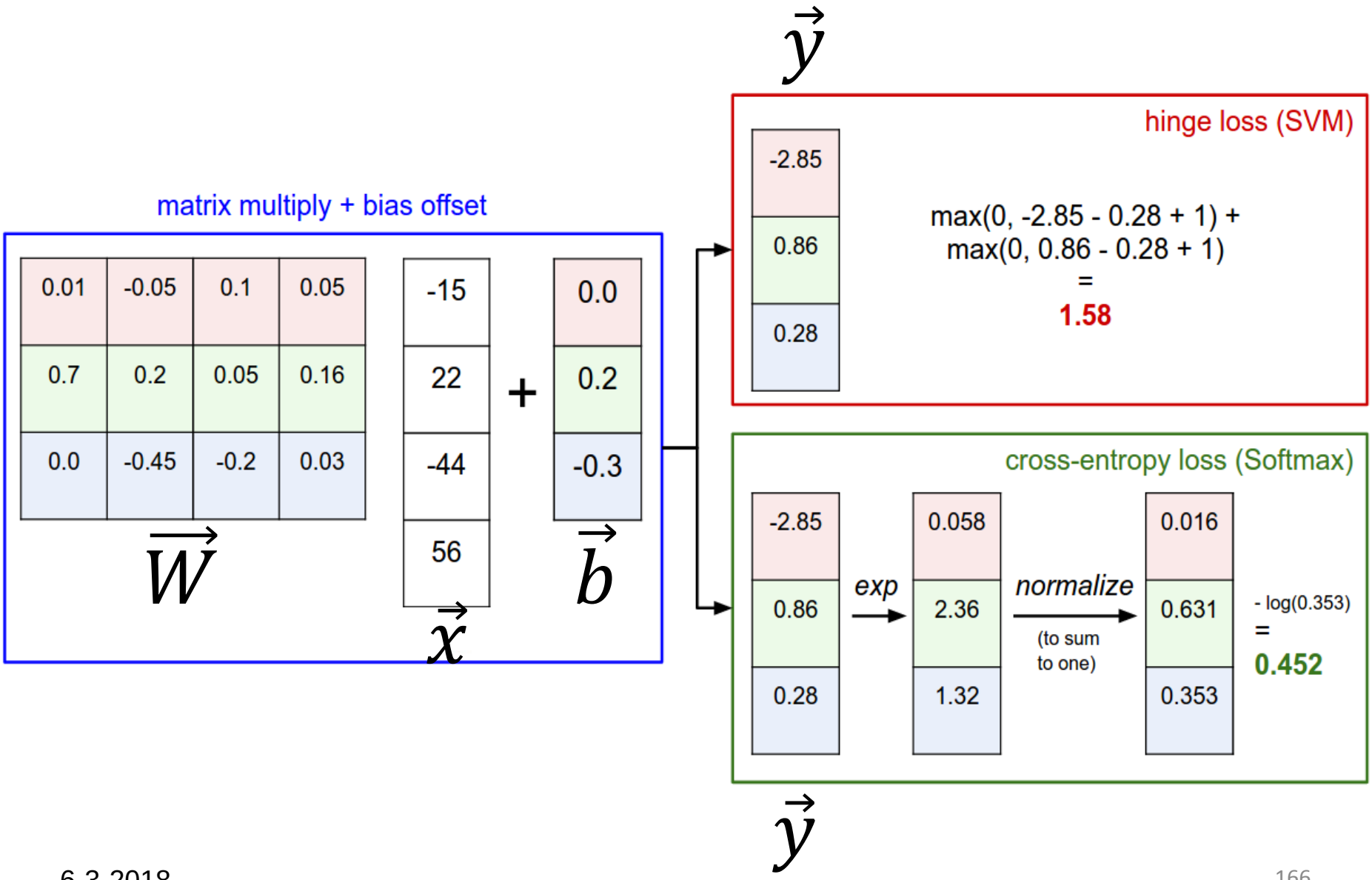
Output=3x1 Weights=3x4 Bias=4x1

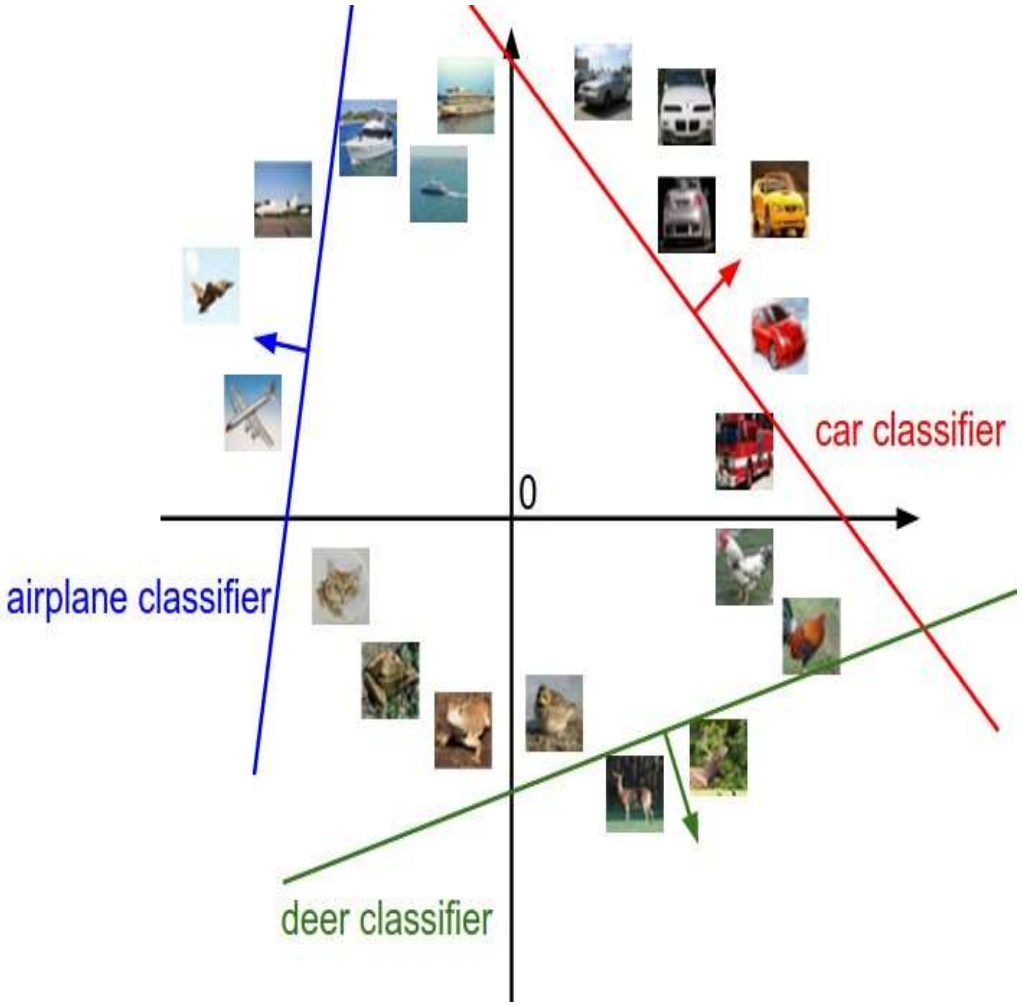
$$\vec{y} = f(\overrightarrow{W} \cdot \vec{x} + \vec{b})$$

Input=4x1

(cat/dog/ship)







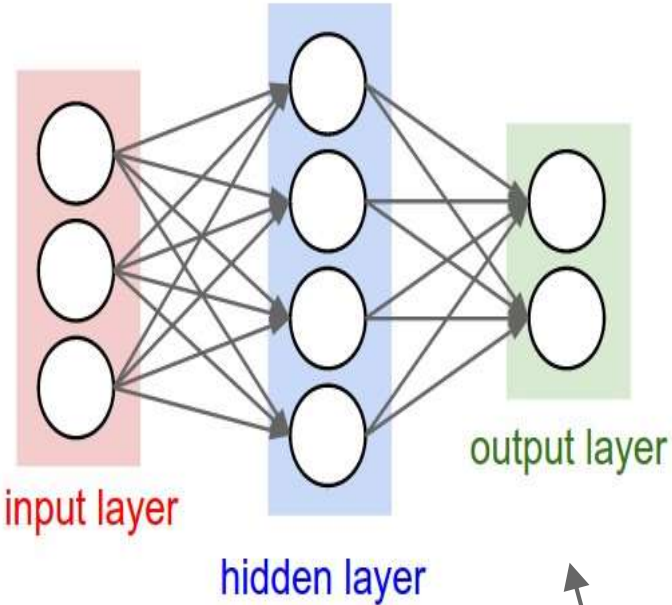
$$\vec{y} = f(\vec{W} \cdot \vec{x} + \vec{b})$$

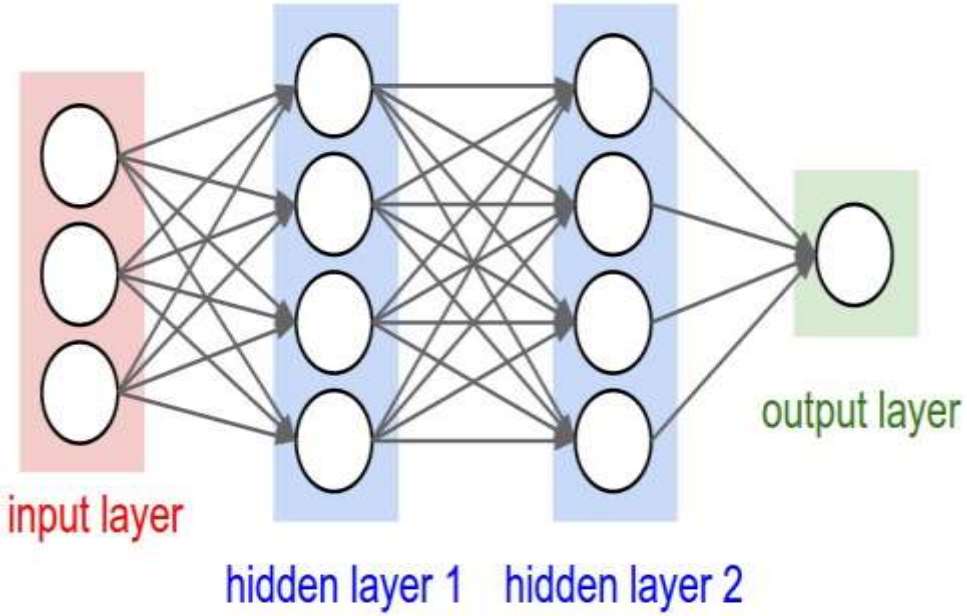
[32x32x3]
array of numbers 0...1
(3072 numbers total)

$$f(x_i, W, b) = Wx_i + b$$

Example trained weights of a linear classifier trained on CIFAR-10:

Andrej Karpathy, Fei-Fei Li, Justin Johnson

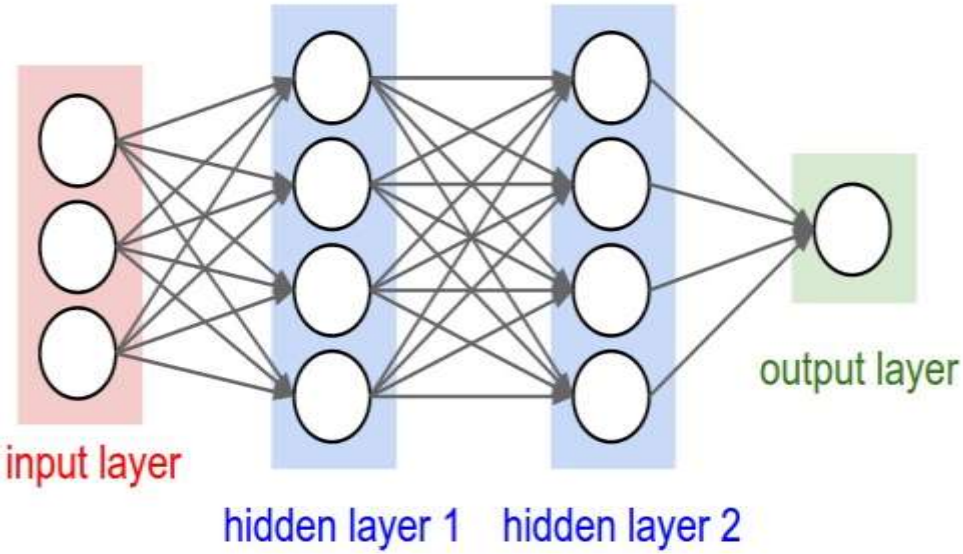




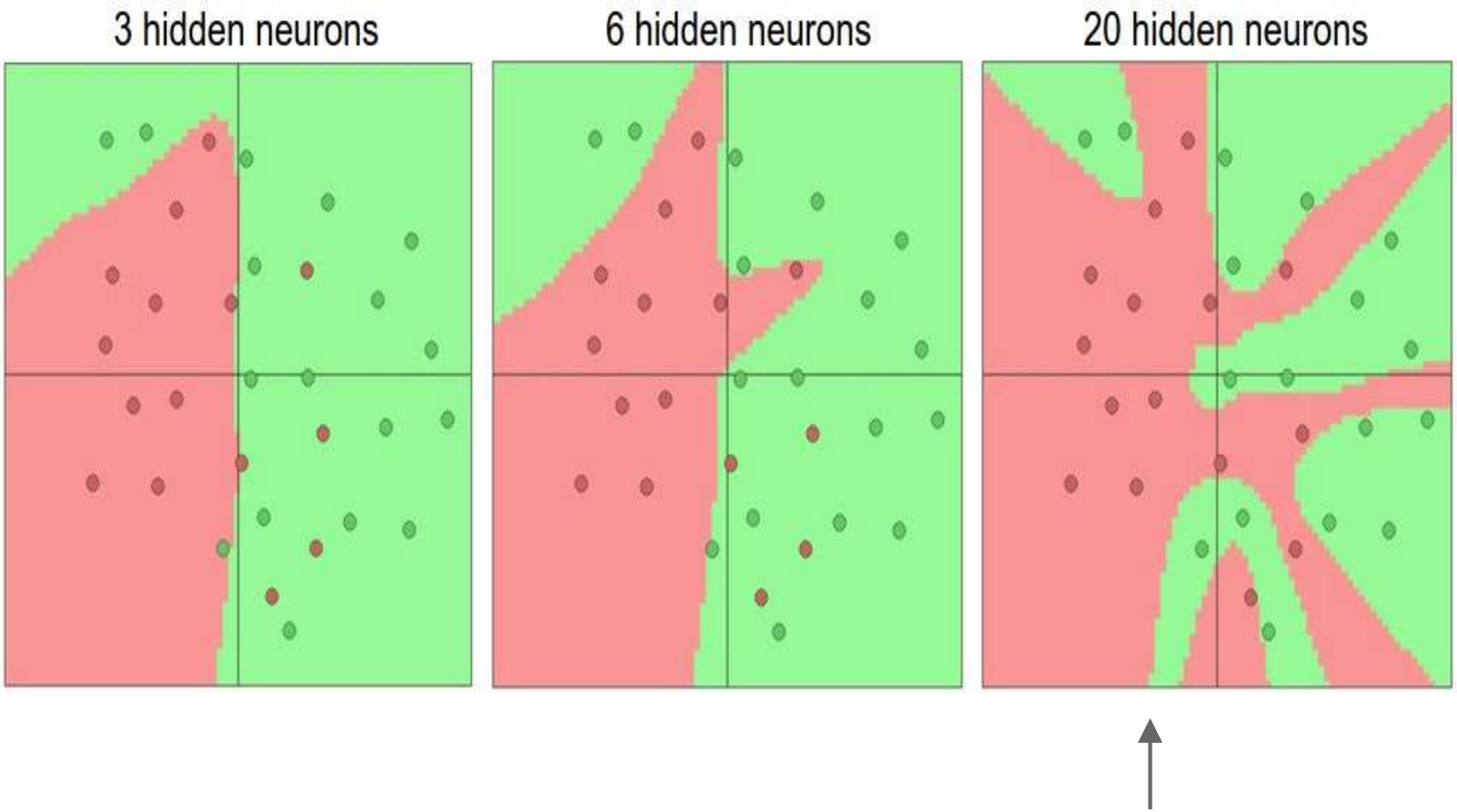
"2-layer Neural Net", or
"1-hidden-layer Neural Net"

"3-layer Neural Net", or
"2-hidden-layer Neural Net"

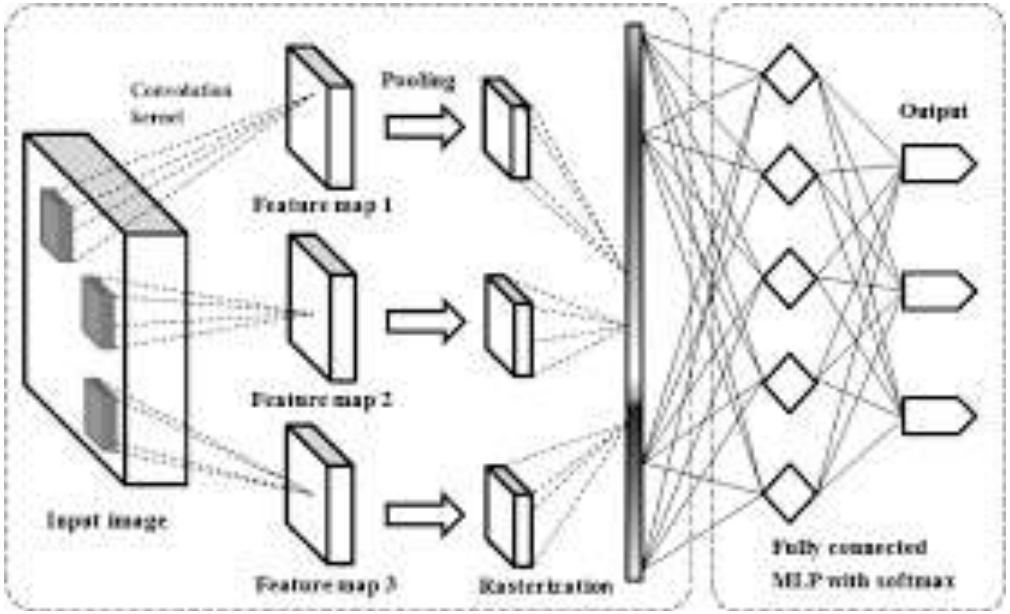
"Fully-connected" layers

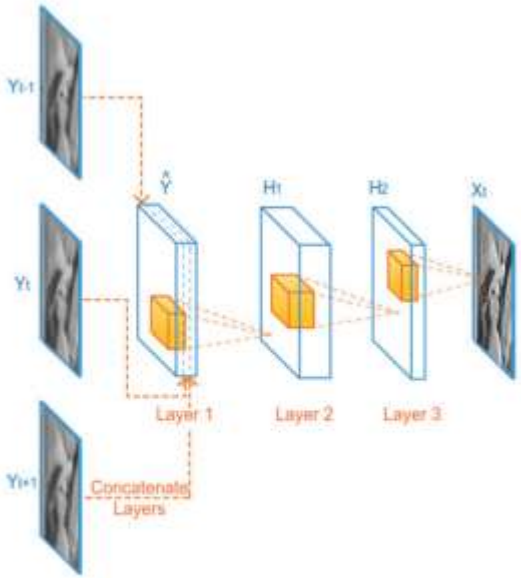



```
# forward-pass of a 3-layer neural network:
f = lambda x: 1.0/(1.0 + np.exp(-x)) # activation function (use sigmoid)
x = np.random.randn(3, 1) # random input vector of three numbers (3x1)
h1 = f(np.dot(W1, x) + b1) # calculate first hidden layer activations (4x1)
h2 = f(np.dot(W2, h1) + b2) # calculate second hidden layer activations (4x1)
out = np.dot(W3, h2) + b3 # output neuron (1x1)
```

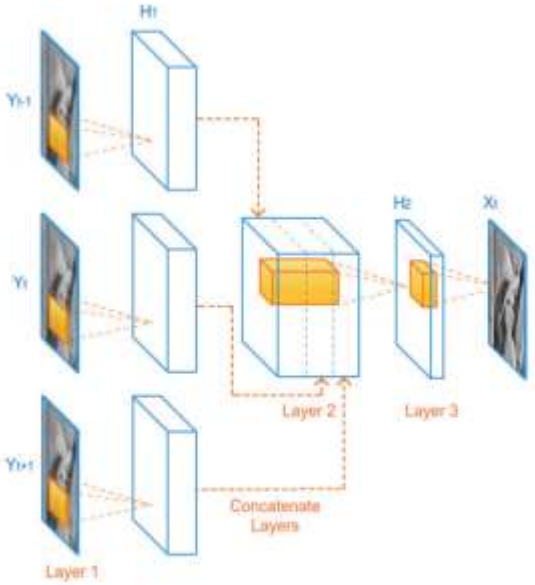

3 hidden neurons
6 hidden neurons
20 hidden neurons
more neurons = more capacity

- we arrange neurons into fully-connected layers
- the abstraction of a **layer** has the nice property that it allows us to use efficient vectorized code (e.g. matrix multiplies)
- neural networks are not really *neural*
- neural networks: bigger = better (but might have to regularize more strongly)

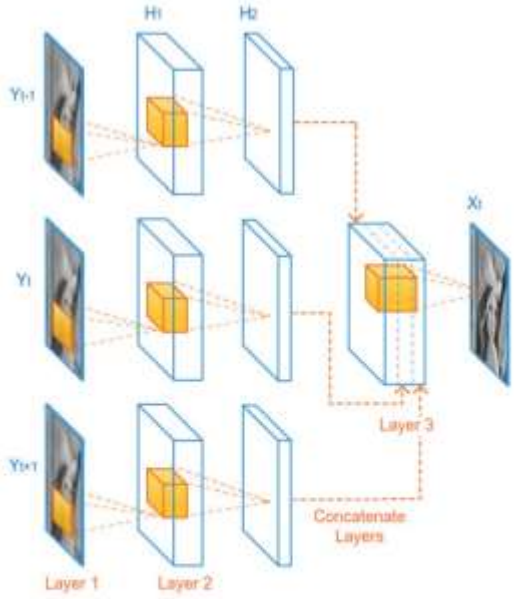




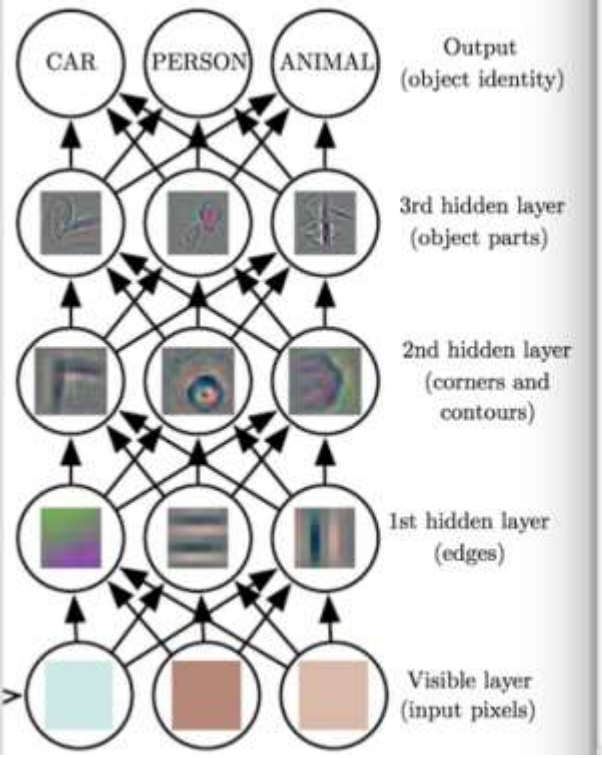
architecture (a)

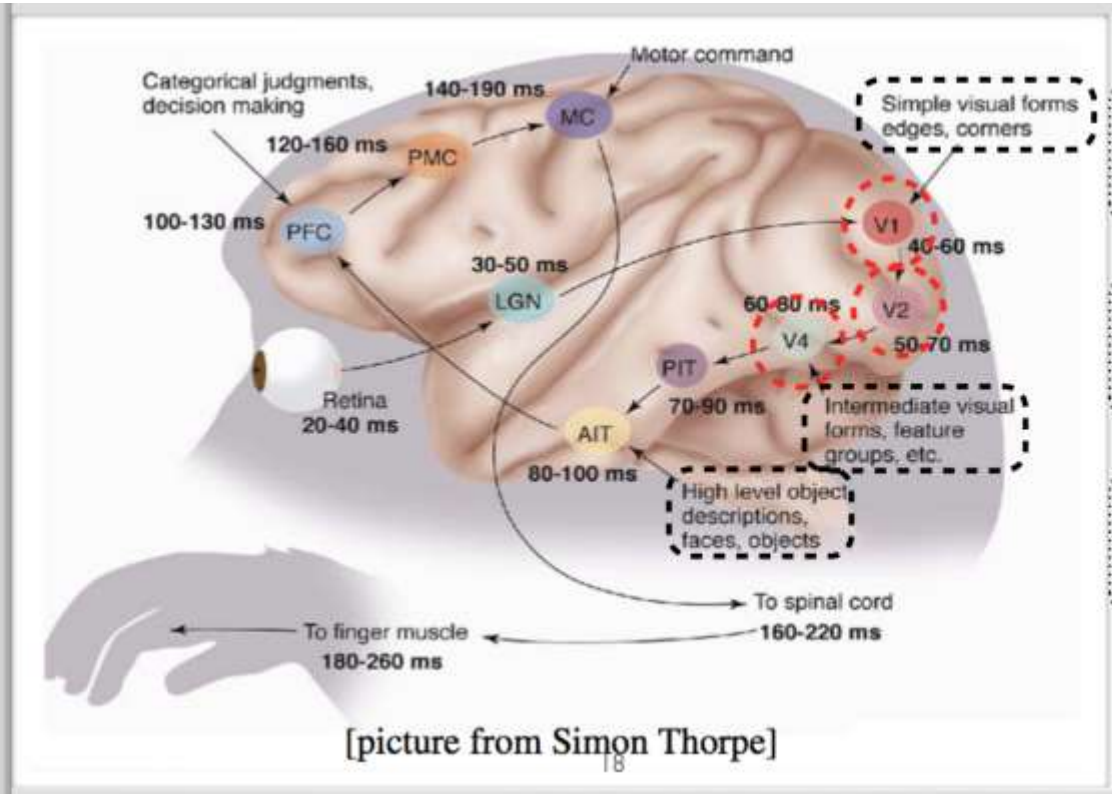


architecture (b)



architecture (c)





[picture from Simon Thorpe]

Computer Vision 1 (total #slides 175 | Lecture 5)

Summary

1. **Segmentation**
2. **Object Recognition**
3. **Bag-of-Words**
4. **Deep Learning**