

Object Recognition (in Automotive Environment)

V-Charge Summer School



Paolo Medici
medici@vislab.it

Outline

- ❑ Introduction on classification
- ❑ Overfitting!
- ❑ Techniques (LDA, PCA, SVM, Boosting, Cascades)
- ❑ Features (Haar, HOG and Integral Channel)
- ❑ Performance Evaluation

Introduction

- ❑ What does it mean «Classify»?
- ❑ What we call object recognition may also be called pattern recognition.
- ❑ A pattern is an arrangement of descriptors.
- ❑ Descriptors may have more forms, but they are vectors or strings.

Timeline of recognition

- ❑ 1965-late 1980s: alignment, geometric primitives
- ❑ 1980's: ANNs come and go; shift toward geometry and increased mathematical rigor
- ❑ 1990's: face recognition; statistical analysis in vogue
- ❑ Early 1990s: invariants, appearance-based methods
- ❑ Mid-late 1990s: sliding window approaches
- ❑ Late 1990s: feature-based methods
- ❑ Early 2000s: parts-and-shape models, large annotated datasets available; video processing starts
- ❑ 2003 – present: bags of features, combination of local and global methods, modeling context, emphasis on “image parsing”

Overview of Recognition

- ❑ 2D Object Recognition vs 3D Object Recognition
- ❑ Window-level detection vs Region Based Segmentation
- ❑ Object orientation and actions recognition
- ❑ Monolithic vs Part Based/Patch Based/Bag of Words

Object Detection

□ Approaches

- Holistic detection: use local search window that meets criteria
- Part-based detection: object as a collection of parts
- Patch-based detection: local features matched against a learned codebook, then voting for final detection

- For a survey on the approaches see: N.Dalal "Finding people in images and videos", PhD thesis, July 2006.

Holistic approaches

- ❑ P. Papageorgiou and T. Poggio, "A trainable system for object detection," Intl. J. of Computer Vision, vol. 38, no. 1, pp. 15–33, 2000.
 - the popular face detector from Viola Jones (haar+adaboost face-detector) + motion cues
- ❑ P. Viola, M. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," in Proc. IEEE Conf. on Computer Vision and Pattern Recognition, New York, NY, volume 1, 2003, pp. 734–741.
 - Histogram of oriented gradients (HOG)
- ❑ Paul Viola and Michael Jones. Robust real-time object detection. International Journal of Computer Vision, 57(2): pp137-154, 2002.
 - Haar feature + AdaBoost + Cascade
- ❑ N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in Proc. IEEE Conf. on Computer Vision and Pattern Recognition, San Diego, CA, volume 1, 2005, pp. 886–893.
- ❑ Q. Zhu, S. Avidan, M. C. Yeh, and K. T. Cheng, "Fast human detection using a cascade of histograms of oriented gradients," in Proc. IEEE Conf. on Computer Vision and Pattern Recognition, New York, NY, volume 2, 2006, pp. 1491 – 1498.
 - Covariance Descriptors + Boosting
- ❑ O. Tuzel, F. Porikli, and P. Meer, "Pedestrian detection via classification on Riemannian manifolds," To appear in IEEE Trans. Pattern Anal. Machine Intell., 2008.
- ❑ P. Dollár, Z. Tu, P. Perona and S. Belongie, "Integral Channel Features" BMVC 2009.
 - Integral Channel Feature, 2-layer decision tree, softcascade

Challenges

- ❑ Viewpoint
- ❑ Illumination
- ❑ Occlusion
- ❑ Scale
- ❑ Deformation
- ❑ Intra-class variation

Training

□ Training methods:

- **Supervised**: the human supervisor provides the labels *a priori*
- **Unsupervised**: data is clustered by algorithm and labels are provided *a posteriori*

Training

□ Training:

- The system works on a data set (the **training set**) composed by pre-classified samples. Using these samples some variables inside the classifiers are **optimized**.

□ Validation:

- The system operates on a **test set**, composed by pre-classifier samples, in order to give an «objective» evaluation.

The Classifier

- What is a «classifier»?
 - Partitioning function on a n -dimensional manifold
 - Transfer function

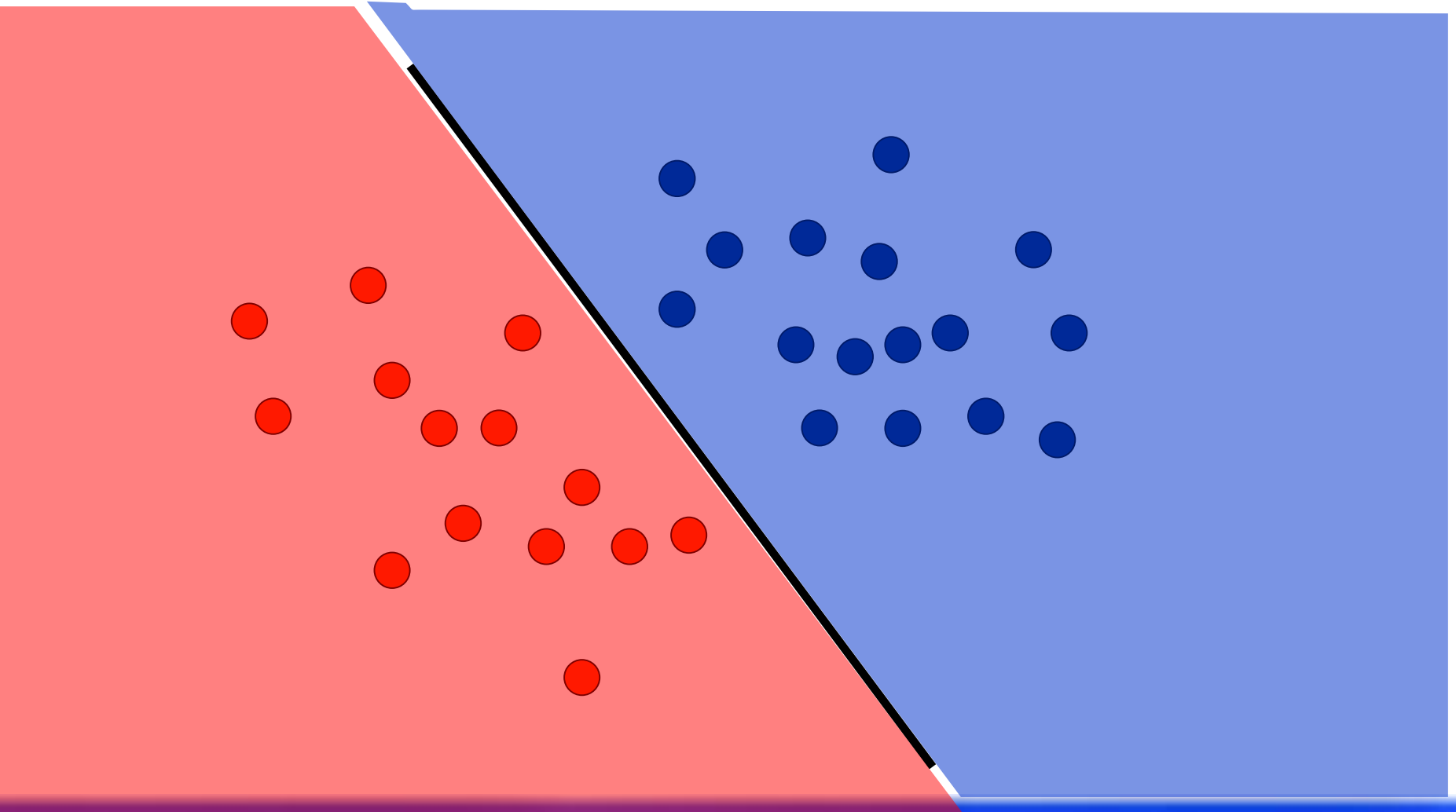
Space Partitioning

$$f : R^n \rightarrow \Omega$$

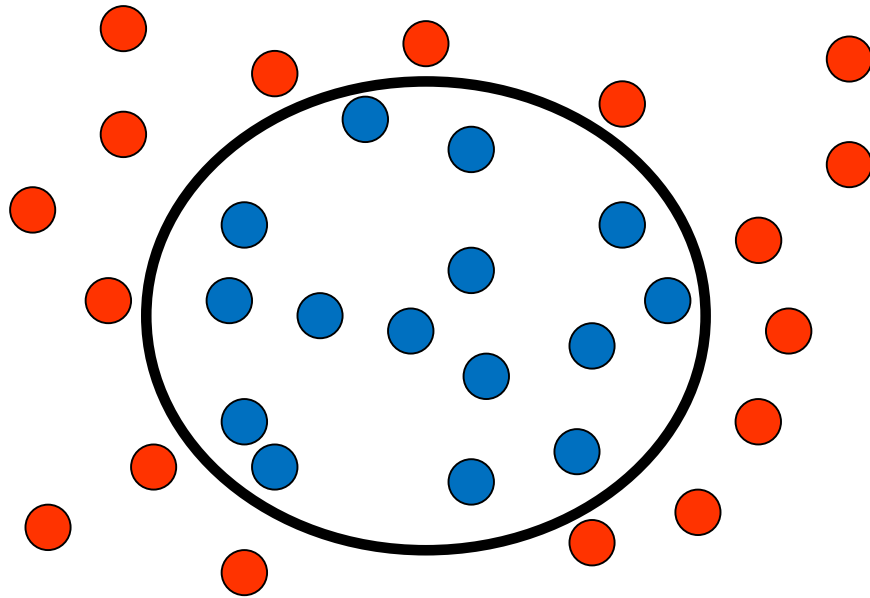
Space Partitioning

$$f : R^2 \rightarrow \{RED, BLUE\}$$

Space Partitioning



Space Partitioning



Space Partitioning

- ❑ N-dimensional space extension
 - Multidimensional manifold
- ❑ Problem: complex and non connected surfaces!
- ❑ Problem: the metric!
- ❑ Tradeoff between computational burden, memory consumption and classification performance

Space Partitioning (the Models)

- ❑ Plane (or better hyperplane)
- ❑ Non linear manifold
- ❑ Decision Tree

Different generalization and performances

Different memory consumption

Different computational burden

Decision Stump/Tree

□ Decision Stump:

□ Decision Tree:

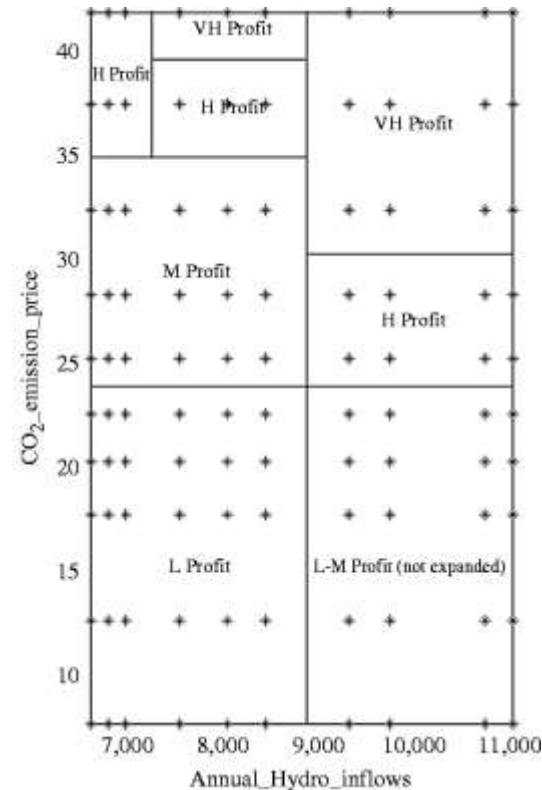
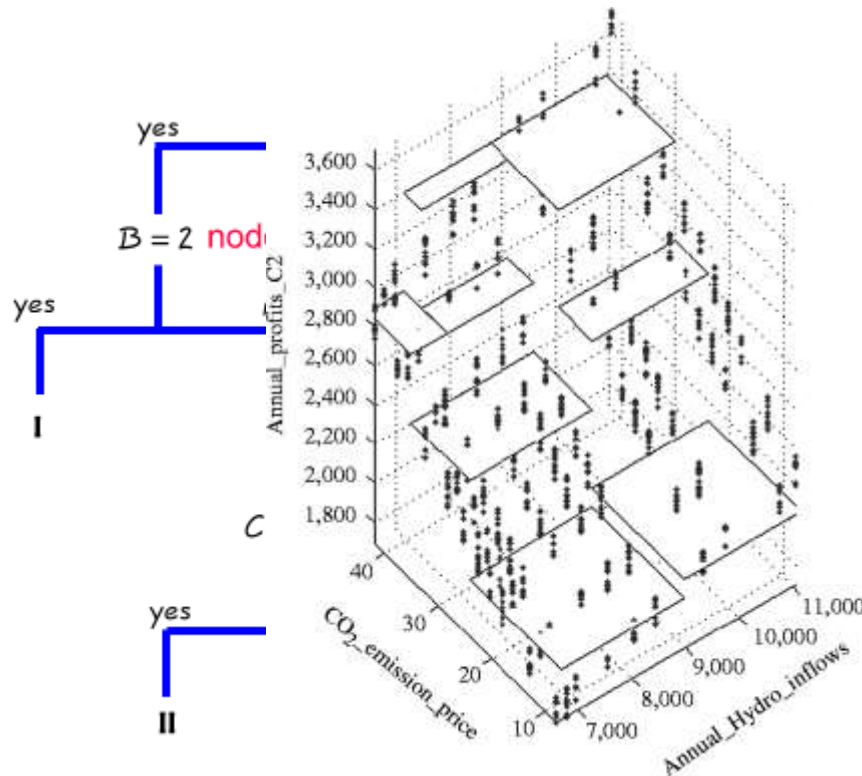
petal width > 1.75

no

yes

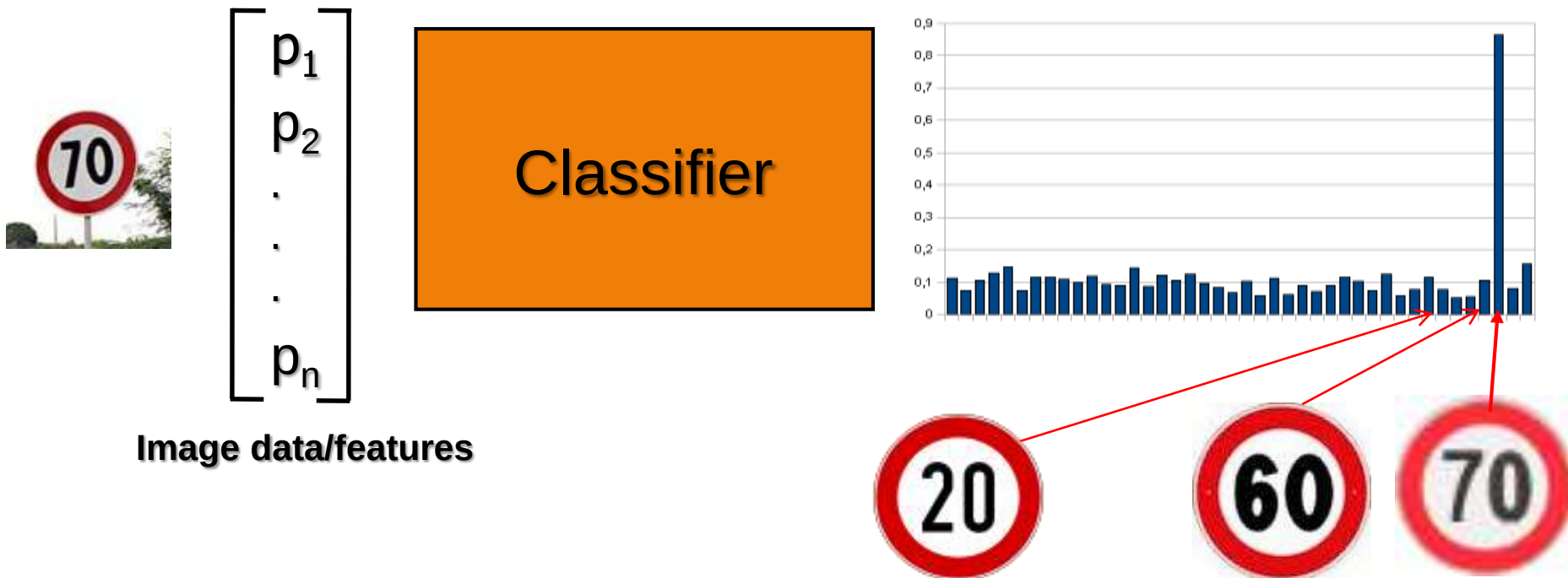
Iris versicolor

Iris virginica



Transfer function

$$f : R^n \rightarrow R^m$$



Training

□ Training

- Space partitions

- Maximize the class separation margin

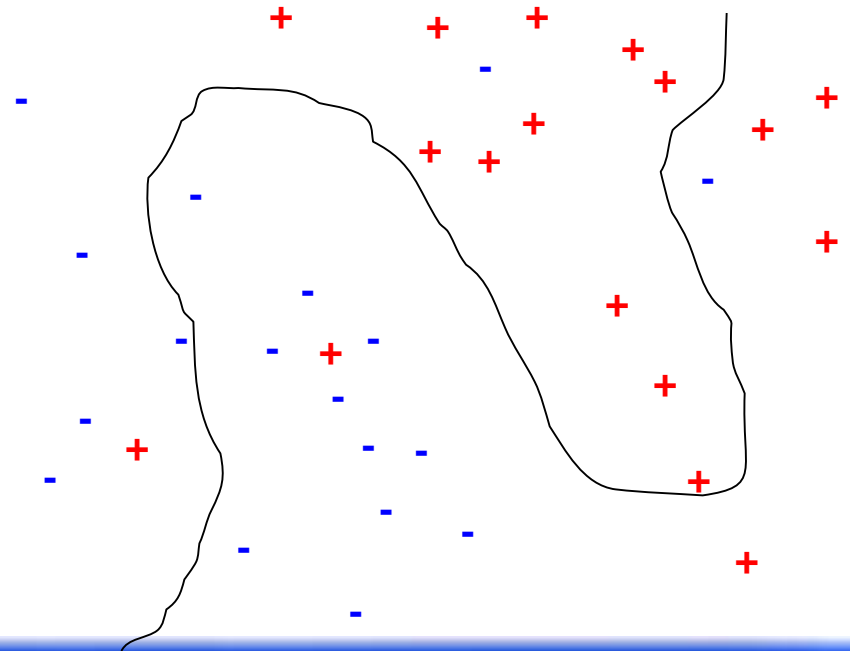
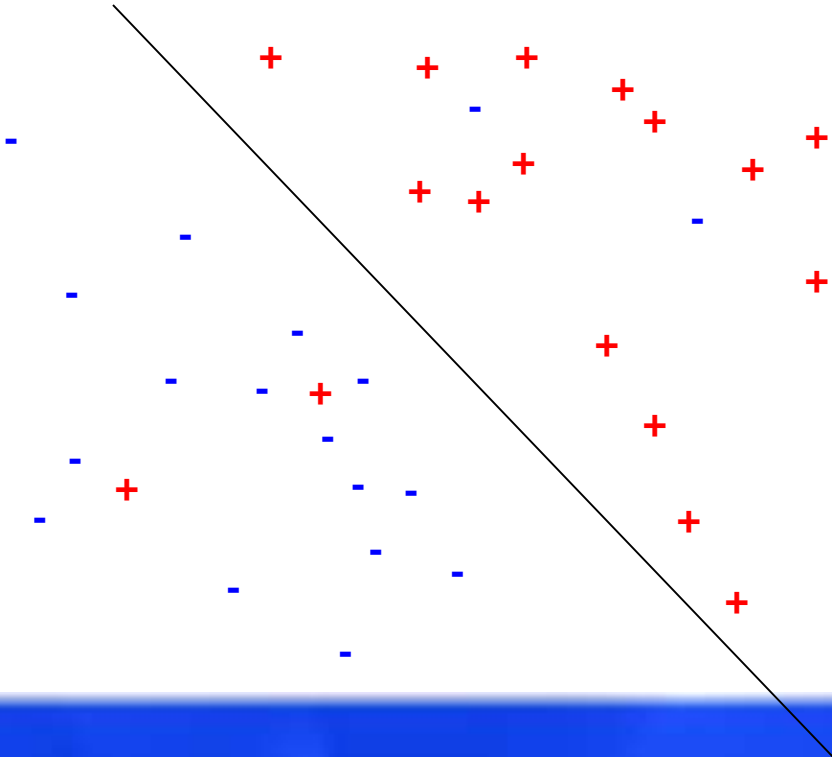
- Transfer function

- Optimize a function, minimizing an error

□ The big problem: **Overfitting**

Overfitting

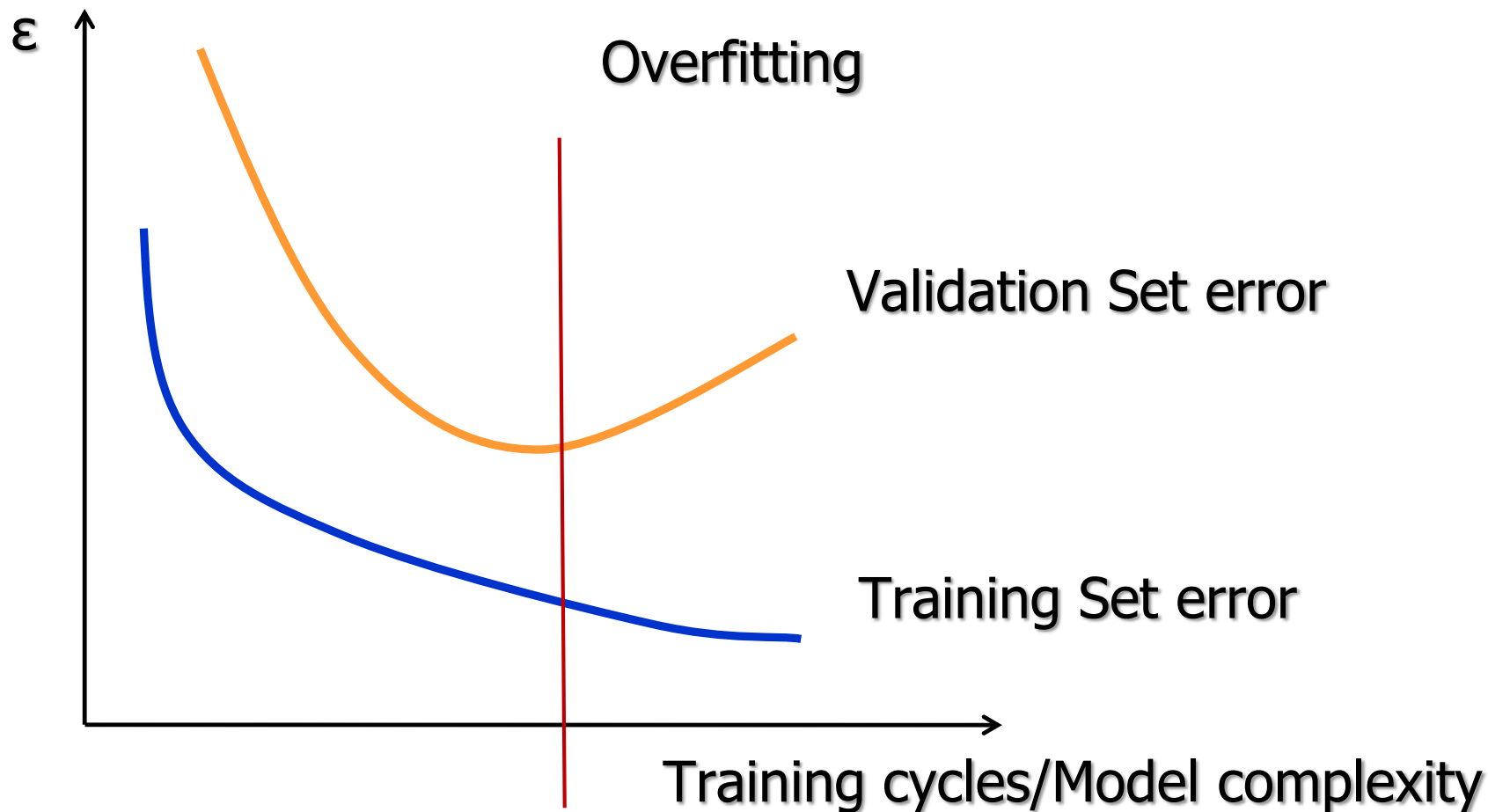
- ❑ Simple model has better generalization.
- ❑ Noise or too much simple model?



Overfitting

- ❑ The model can fit to the training set – specific – features and it is not able to classify new samples
 - Generalization lost
- ❑ Reasons
 - Too much training iteration
 - Too much complex model
 - Small training set

Overfitting



Overfitting and Performance

- ❑ Given a range of classification algorithms, which is the best?
- ❑ Some algorithms may be preferred because of their low complexity, ability to incorporate prior knowledge,....
- ❑ Principle of **Occam's Razor**: given two classifiers that perform equally well on the training set, it is asserted that the simpler classifier may do better on test set

No Free Lunch Theorem

- ❑ Suppose we make no prior assumptions about the nature of the classification task. Can we expect any classification method to be superior or inferior overall?
- ❑ No **Free Lunch Theorem**: Answer to above question: NO
- ❑ If the goal is to obtain good generalization performance, there is no context-independent or usage-independent reasons to favor one algorithm over others
- ❑ If one algorithm seems to outperform another in a particular situation, it is a consequence of its fit to a particular pattern recognition problem
- ❑ For a new classification problem, what matters most: prior information, data distribution, size of training set, cost fn.

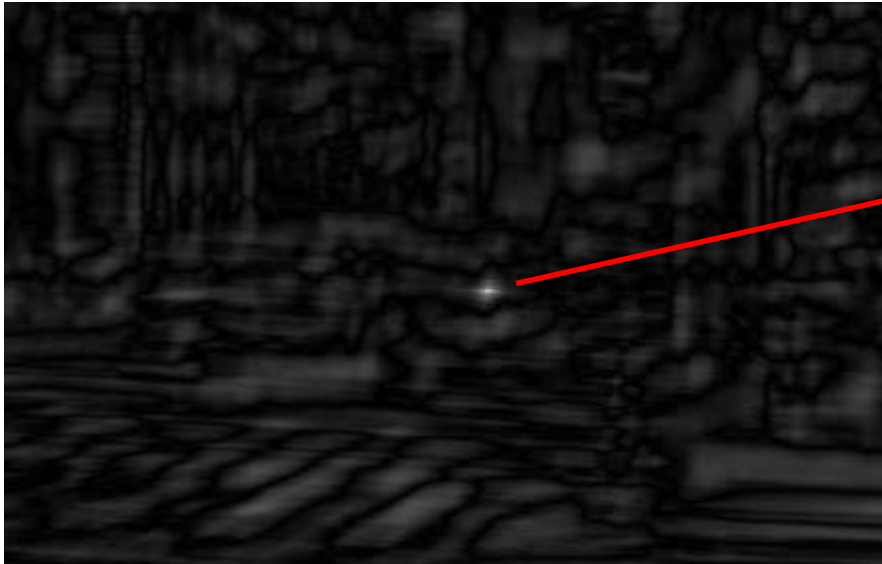
No Free Lunch Theorem

- ❑ It is the assumptions about the learning algorithm that are important
- ❑ Even popular algorithms will perform poorly on some problems, where the learning algorithm and data distribution do not match well
- ❑ In practice, experience with a broad range of techniques is the best insurance for solving arbitrary new classification problems

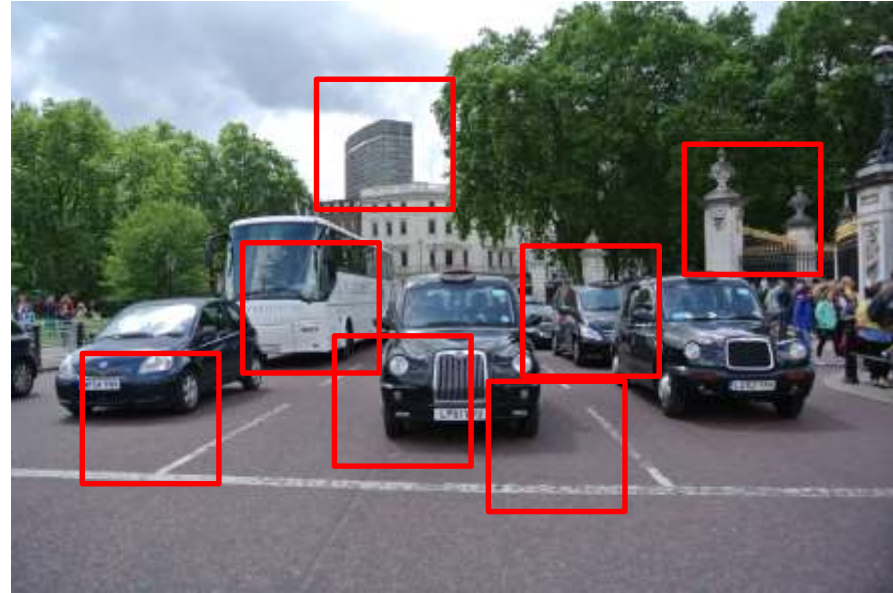
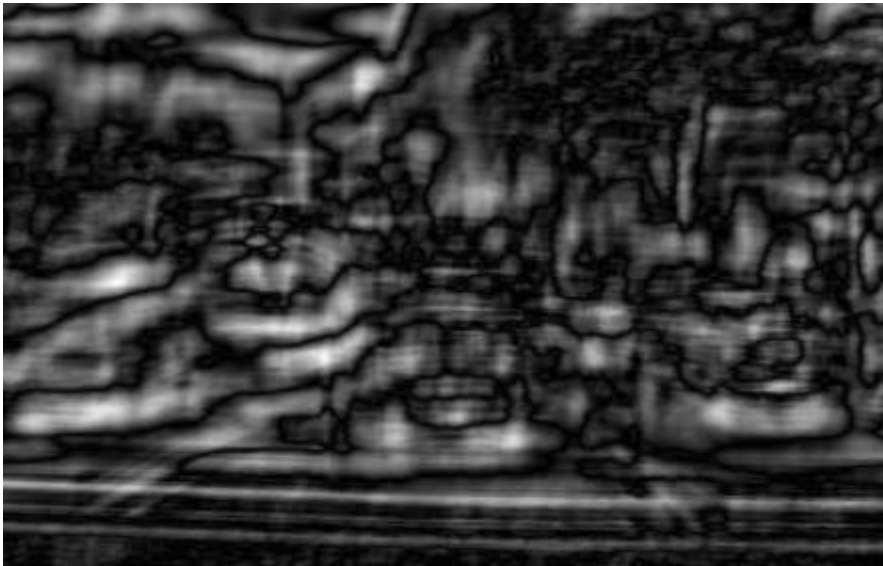
Object Detection

- ❑ To create a good feature-space representation of the problem
- ❑ To develop a good algorithm able to separate the class and a model able to handle this morphological information

Template Matching



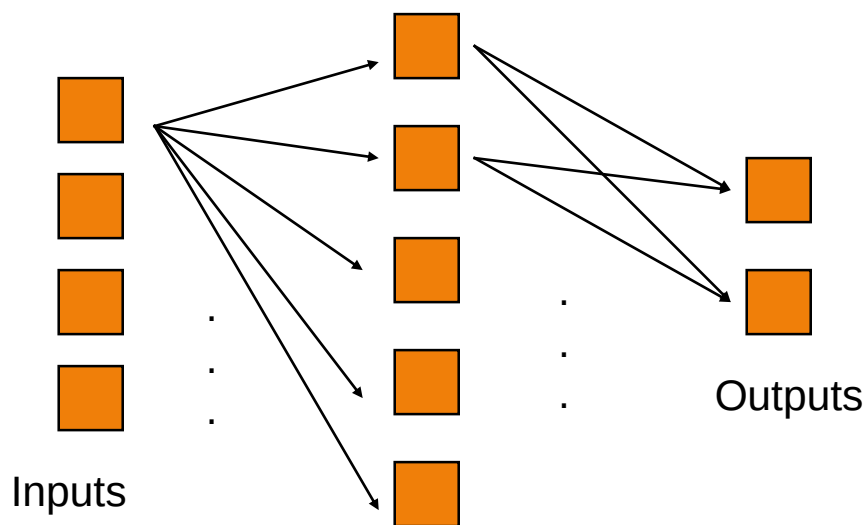
Template Matching



NEW YORK TAXI PATTERN DOES NOT WORK ON LONDON TAXI

Artificial Neural Nets (1980s)

- ❑ Artificial Neural Nets (ANNs) are networks of artificial neuron nodes, each of which computes a simple function.
- ❑ An ANN has an input layer, an output layer, and “hidden” layers of nodes.



Artificial Neural Nets

- Between 2009 and 2012, the recurrent neural networks and deep feedforward neural networks developed in the research group of Jürgen Schmidhuber at the Swiss AI Lab IDSIA have won eight international competitions in pattern recognition and machine learning.

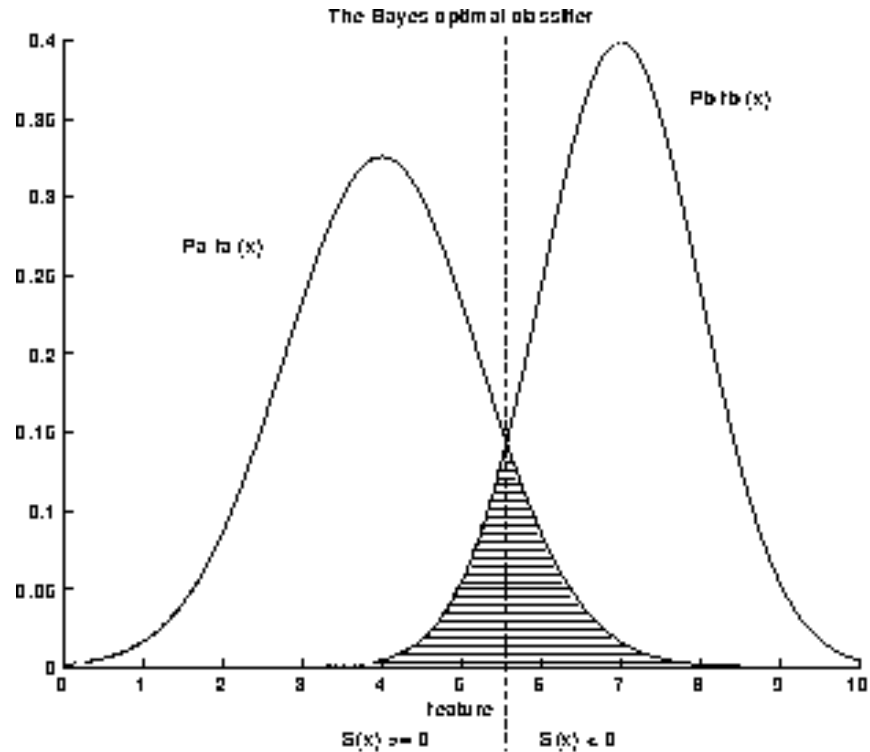
Bayes Classifiers

□ Bayes Classifier

$$\frac{P(x / \text{Object})}{P(x / \text{nonObject})}$$

□ Naive

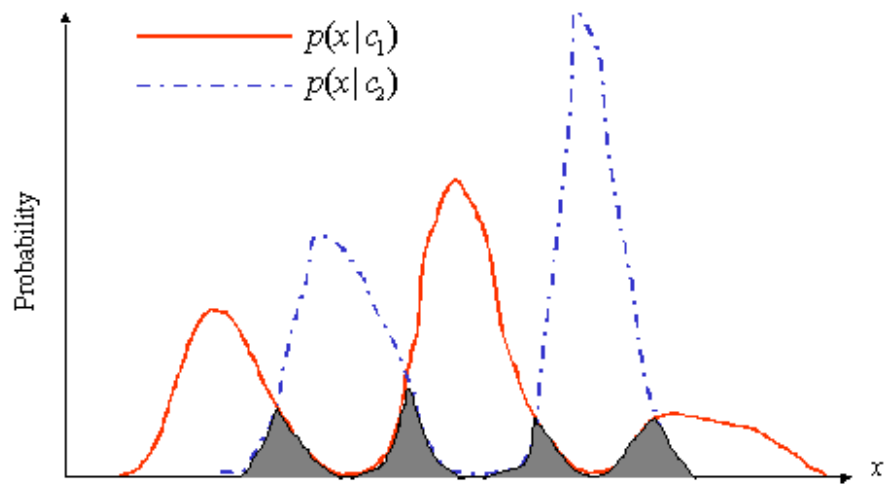
$$\prod_{i=1}^K \frac{P(F_i(x) / \text{Object})}{P(F_i(x) / \text{nonObject})} \geq \lambda$$



Bayes Classifiers

□ Training Procedure:

- Estimate $P(F_k(x) / Object)$ and $P(F_k(x) / nonObject)$ using a parametric model or using histograms.
- Each of the histograms represents the $F_k(x)$ statistic of appearance given by

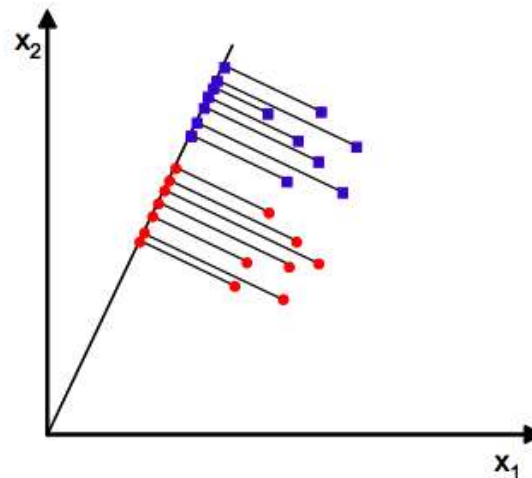
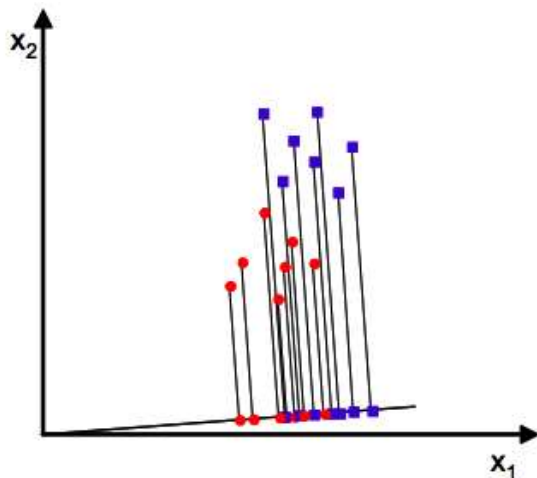


Linear Classifiers

- Positive and negative examples are separated by a linear function (hyperplane)

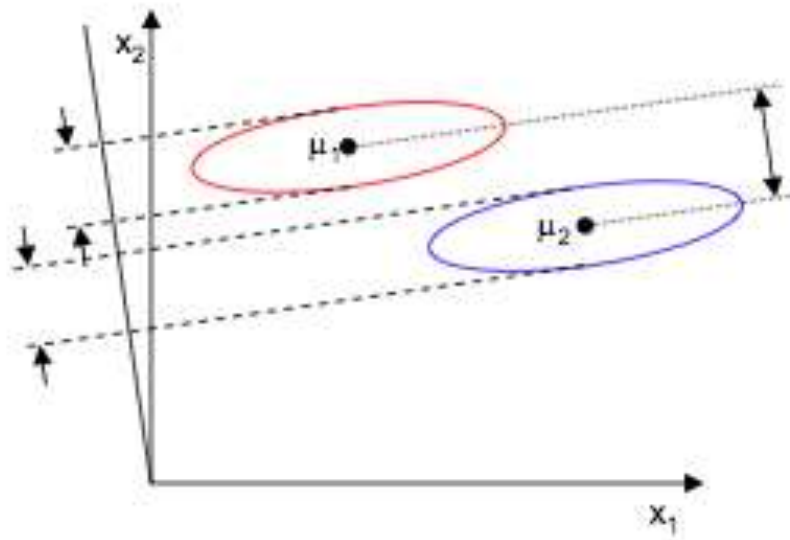
Linear Discriminant Analysis

- ❑ Model the classifier using a linear combination of features value
 - Reduce the dimensional space
 - Maximize the class separation
- ❑ Example in 2-feature space:
 - Select the projection line that separates the classes



Linear Discriminant Analysis

- To detect the projection vector it is mandatory define a separation metric:
 - Separate the mean values
 - Consider also the standard deviation



Linear Discriminant Analysis

- Recover $\mathbf{w}$, the linear application $\mathbf{w}^T \mathbf{x}$, that maximize:

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}}$$

- where $\mathbf{S}_B = \sum_{i=1}^C N_i (\mu_i - \mu)(\mu_i - \mu)^T$ Between-class scatter matrix

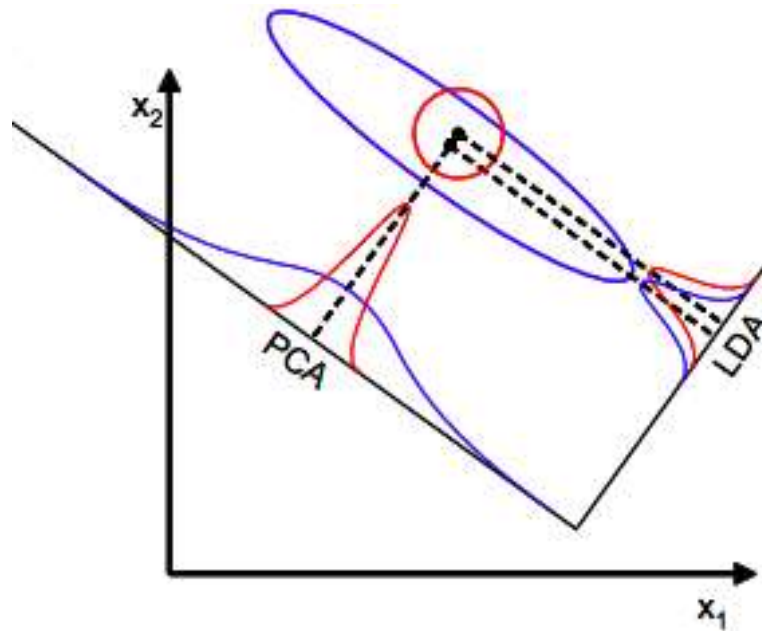
$$\mu = \frac{1}{N} \sum_{\forall \mathbf{x}} \mathbf{x} = \frac{1}{N} \sum_{\mathbf{x} \in \omega_i} N_i \mu_i$$

$$\mathbf{S}_W = \sum_{i=1}^C \mathbf{S}_i \quad \text{Within-class scatter matrix}$$

$$\mathbf{S}_i = \sum_{\mathbf{x} \in \omega_i} (\mathbf{x} - \mu_i)(\mathbf{x} - \mu_i)^T \quad \mu_i = \frac{1}{N_i} \sum_{\mathbf{x} \in \omega_i} \mathbf{x}$$

LDA: limits

- Sometimes it is not the average but the variance the discrimination element



Principal Component Analysis

- ❑ La Principal Component Analysis (Karhunen-Loeve) exploits orthogonal transformation in order to convert correlated variables in uncorrelated, called principal component.
- ❑ A ranking!
 - The first component has the maximum variance
 - The components are ordered using variance and they are orthogonal bases of space

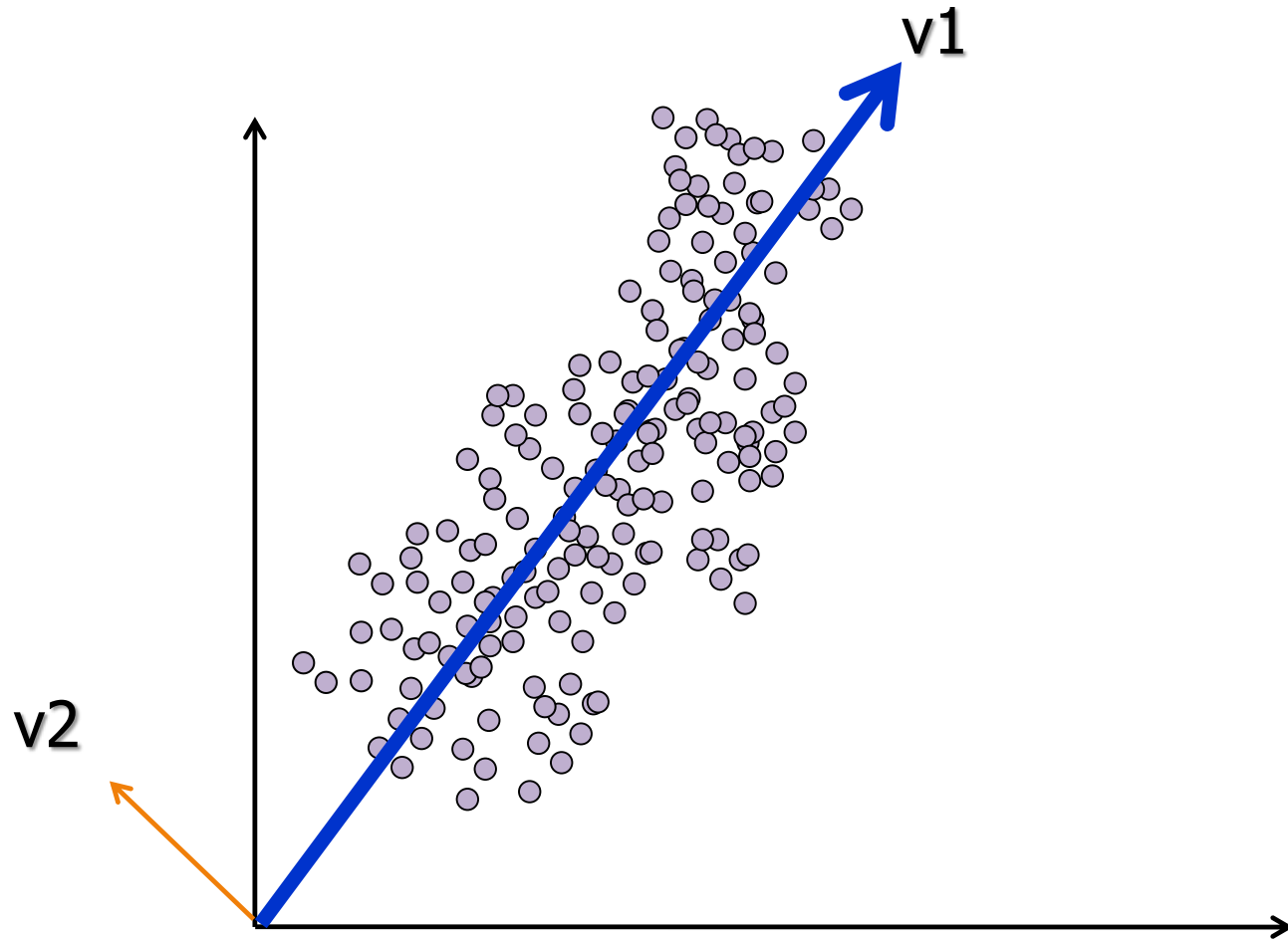
Principal Component Analysis

- The feature response matrix **A** is decomposed (using SVD)

$$\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T \quad (\text{SVD})$$

- **V** contain the Eigenvectors, ranked according to Eigenvalues value and, consequently, the variance.
- It is possible to create a subspace $\mathbf{V} \sim$ with less elements compared to **V**
- Using the first columns of **V**, most of the information is preserved

Principal Component Analysis



PCA: Example

$$A = \begin{pmatrix} -9.9932 & -5.0021 \\ -7.9943 & -3.994 \\ -5.9918 & -3.006 \\ -4.0033 & -1.9946 \\ -2.0044 & -0.99892 \\ -0.00045206 & 0.0025774 \\ 1.9973 & 1.0003 \\ 4.009 & 2.0083 \\ 6.0027 & 3.0043 \\ 7.9928 & 4.0021 \end{pmatrix}$$

$$S = \begin{pmatrix} 20.61 \\ 0.013079 \end{pmatrix}$$

$$V = \begin{pmatrix} 0.89428 & 0.44751 \\ 0.44751 & -0.89428 \end{pmatrix}$$

$$V\sim = \begin{pmatrix} 0.89428 \\ 0.44751 \end{pmatrix}$$

$$X = \begin{pmatrix} -9.9932 & -5.0021 \end{pmatrix}$$

$$Y = V\sim X = -11.1$$

Principal Component Analysis

□ Limits

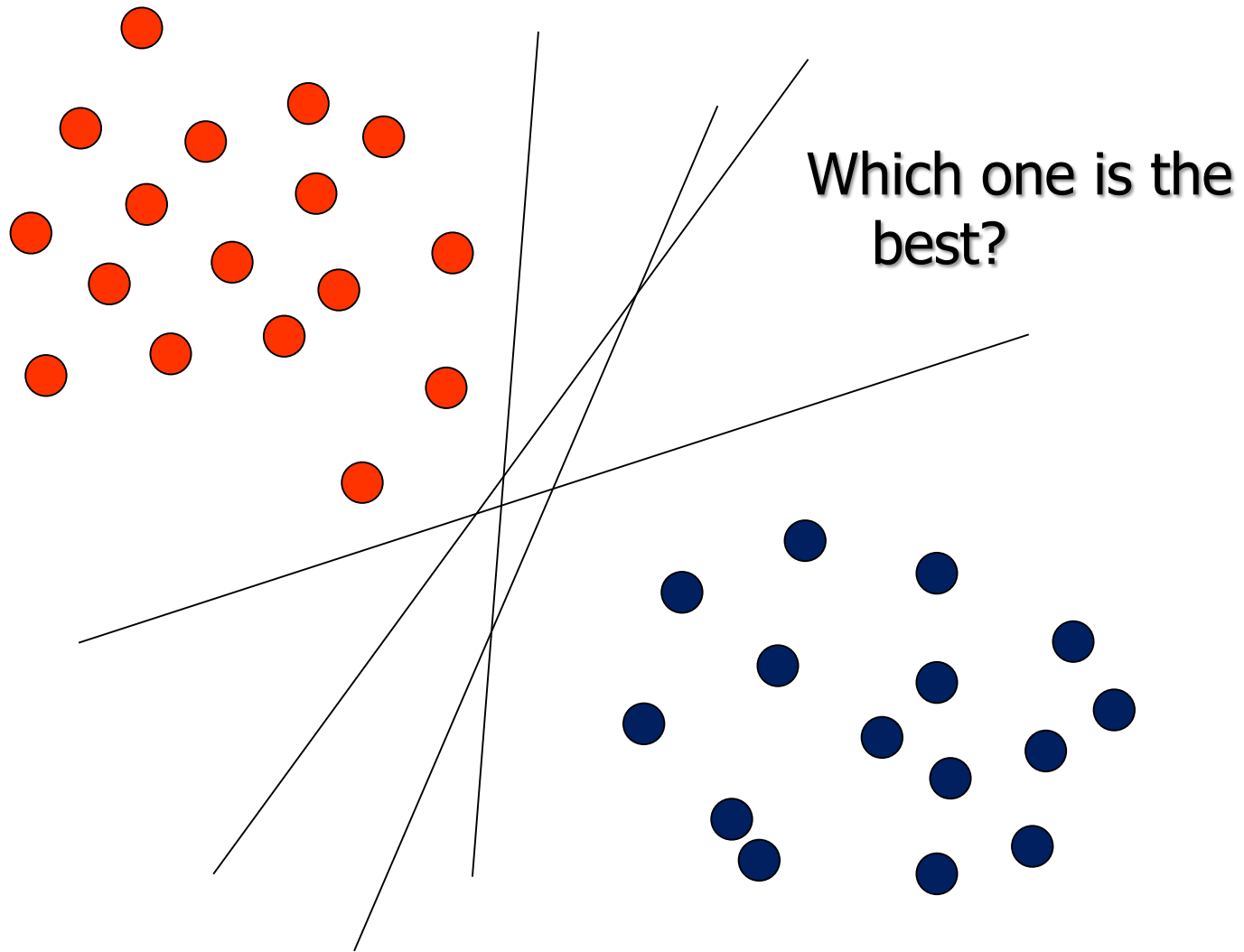
- Does not consider the labels (it is an **unsupervised** learning!)
- There is no guarantee that the maximum variance contains the most discriminant features

Eigenfaces (Turk & Pentland, 1991)

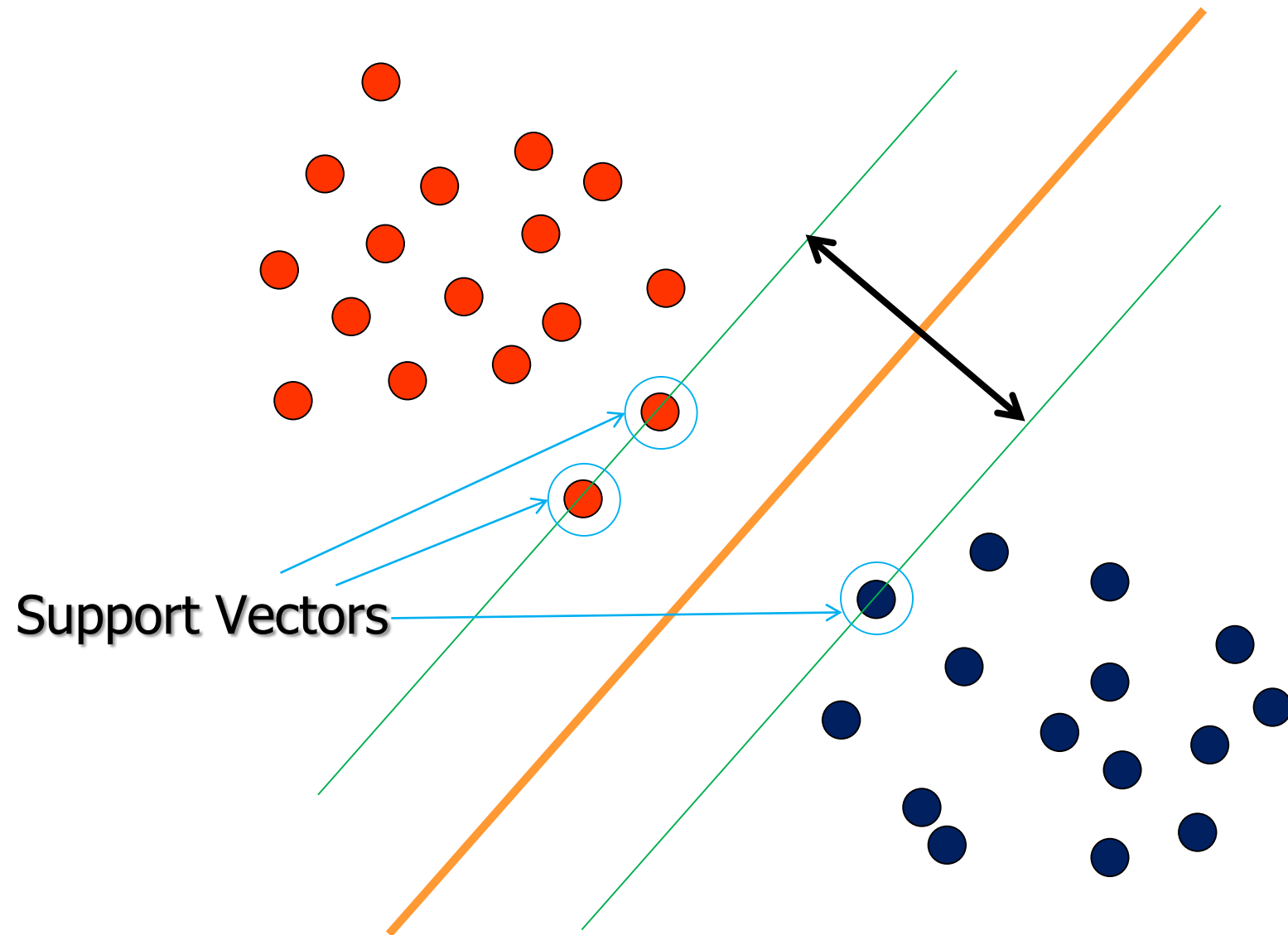


Experimental Condition	Correct/Unknown Recognition Percentage		
	Lighting	Orientation	Scale
Forced classification	96/0	85/0	64/0
Forced 100% accuracy	100/19	100/39	100/60
Forced 20% unknown rate	100/20	94/20	74/20

Support Vector Machines



Support Vector Machines



Summary: Support Vector Machine

- ❑ Large Margin Classifier
 - Better generalization ability & less over-fitting
- ❑ The Kernel Trick
 - Map data points to higher dimensional space in order to make them linearly separable.
 - Since only dot product is used, we do not need to represent the mapping explicitly.

Enforced learning

- It is possible to combine together multiple weak classifier in order to obtain a stronger one?

Michael Kearns, 1988

Weak classifier

- ❑ A classifier that not reach 100%, but, at least, 50%+1
- ❑ Ensemble Learning: merge together weak classifier (it is not a classifier itself, but a technique to suggest how to merge the classifier)

Boosting (R. Schapire, 1990)

- ❑ Boost the performance of a weak learner to the level of a strong one
- ❑ Boosting creates an ensemble of classifiers by resampling the data; classifiers combined by majority voting
 - resampling is strategically geared to provide the most informative training data for each consecutive classifier
- ❑ Boosting creates three weak classifiers:
 - First classifier C1 is trained with a random subset of the available training data
 - Training set for second classifier C2 is chosen as the most informative subset, given C1; half of the training data for C2 is correctly classified by C1, other half is misclassified by C1
 - Third classifier C3 is trained on instances on which both C1 & C2 disagree

Ensemble-based Systems in Decision Making

- ❑ For many tasks, we often seek second opinion before making a decision, sometimes many more
 - Consulting different doctors before a major surgery
 - Reading reviews before buying a product
 - Requesting references before hiring someone
- ❑ We consider decisions of multiple experts in our daily lives. Do we give the same weight at any expert? (e.g. mum)
- ❑ Why not follow the same strategy in automated decision making?
- ❑ Multiple classifier systems, committee of classifiers, mixture of experts, ensemble based systems

Why Ensemble Based Systems?

❑ Statistical reasons

- A set of classifiers with similar training performances may have different generalization performances
- Combining outputs of several classifiers reduces the risk of selecting a poorly performing classifier

❑ Large volumes of data

- If the amount of data to be analyzed is too large, a single classifier may not be able to handle it; train different classifiers on different partitions of data

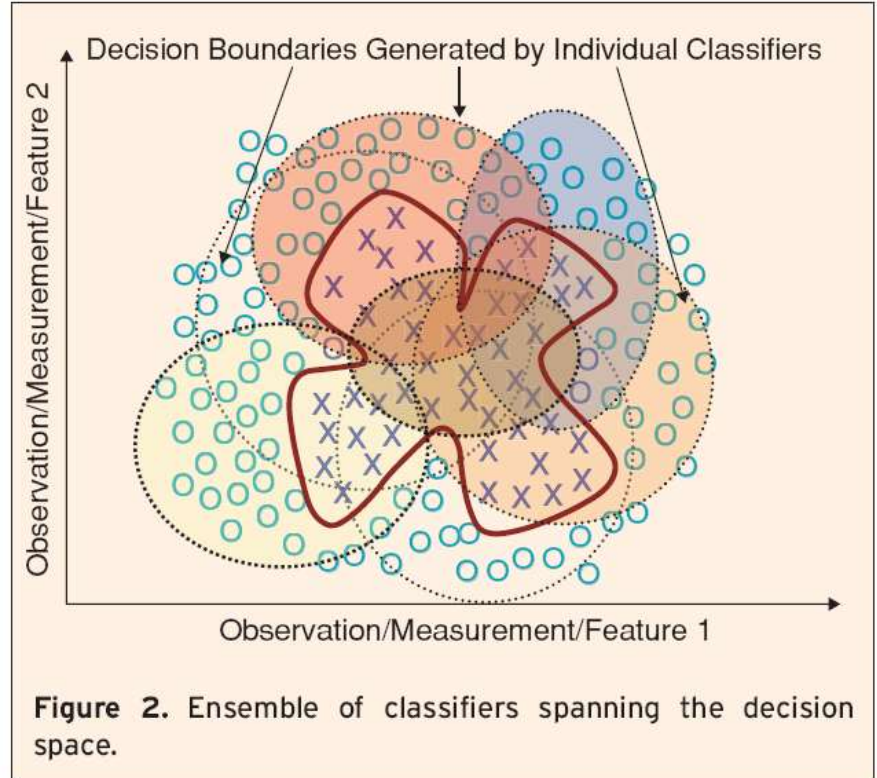
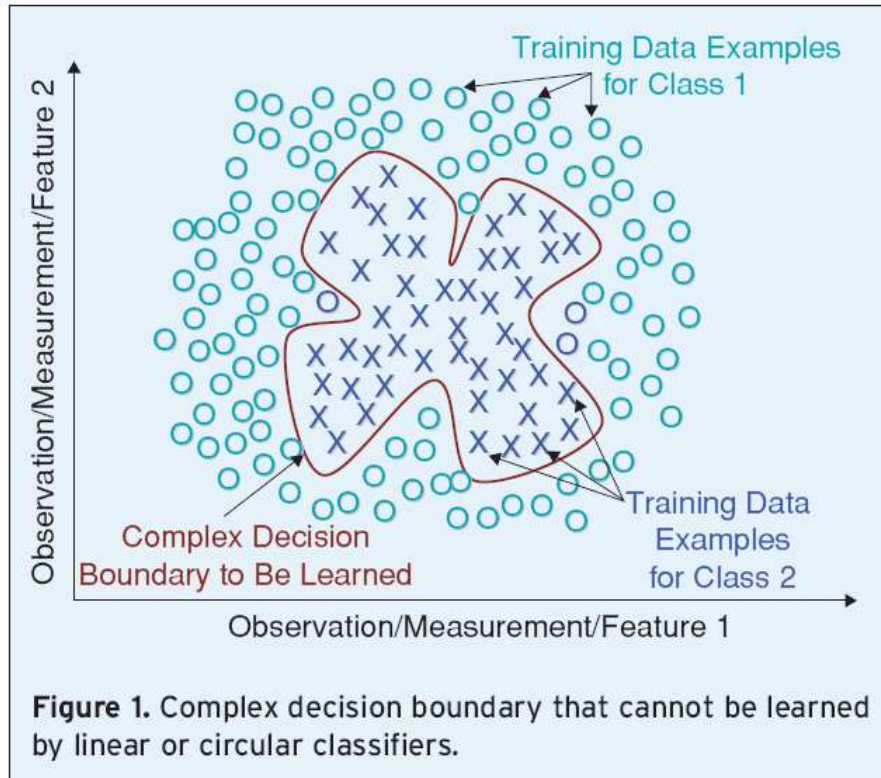
❑ Too little data

- Ensemble systems can also be used when there is too little data; resampling techniques

Why Ensemble Based Systems?

□ Divide and Conquer

- Divide data space into smaller & easier-to-learn partitions; each classifier learns only one of the simpler partitions



Brief History of Ensemble Systems

- ❑ Dasarathy and Sheela (1979) partitioned the feature space using two or more classifiers
- ❑ Schapire (1990) proved that a strong classifier can be generated by combining weak classifiers through boosting; predecessor of AdaBoost algorithm
- ❑ Two types of combination:
 - classifier selection
 - Each classifier is trained to become an expert in some local area of the feature space; one or more local experts can be nominated to make the decision
 - classifier fusion
 - All classifiers are trained over the entire feature space; fusion involves merging the individual (weaker) classifiers to obtain a single (stronger) expert of superior performance

Diversity of Ensemble

- ❑ Objective: create many classifiers, and combine their outputs to improve the performance of a single classifier
- ❑ Intuition: if each classifier makes different errors, then their strategic combination can reduce the total error!
- ❑ Need base classifiers whose decision boundaries are adequately different from those of others
 - Such a set of classifiers is said to be diverse
- ❑ How to achieve classifier diversity?
 - Use different training sets to train individual classifiers
 - How to obtain different training sets?
 - Resampling techniques: bootstrapping or bagging, training subsets are drawn randomly, usually with replacement, from the entire training set

Creating An Ensemble

- Two questions:
 - How will the individual classifiers be generated?
 - How will they differ from each other?
- Answer determines the diversity of classifiers & fusion performance
- Seek to improve ensemble diversity by some heuristic methods

Bagging (Leo Breiman, 1996)

- ❑ **Bagging**, short for **bootstrap aggregating**, is one of the earliest ensemble based algorithms
- ❑ One of the most intuitive and simplest to implement, with a surprisingly good performance
- ❑ The training set is splitted (with replica) in a very large number of training subset
- ❑ Each resampled training set is used to train a different classifier of the same type
- ❑ Individual classifiers are combined by taking a majority vote of their decisions

Variations of Bagging

□ Random Forests™

- so-called because it is constructed from **decision trees**
- A random forest is created from individual decision trees, whose training parameters vary randomly
- Such parameters can be **bootstrapped replicas** of the training data, as in bagging
- But they can also be **different feature subsets** as in random subspace methods

AdaBoost (Freund&Schapire, 1997)

- ❑ AdaBoost is a more general version of the boosting algorithm; AdaBoost.M1 can handle multiclass problems
- ❑ AdaBoost generates a set of hypotheses (classifiers), and combines them through **weighted majority** voting of the classes predicted by the individual hypotheses
- ❑ Hypotheses are generated by training a weak classifier; samples are drawn from an iteratively updated distribution of the training set
- ❑ This distribution update ensures that instances misclassified by the previous classifier are more likely to be included in the training data of the next classifier
- ❑ Consecutive classifiers are trained on increasingly hard-to-classify samples

*Boost

- Defines a classifier using an additive model:

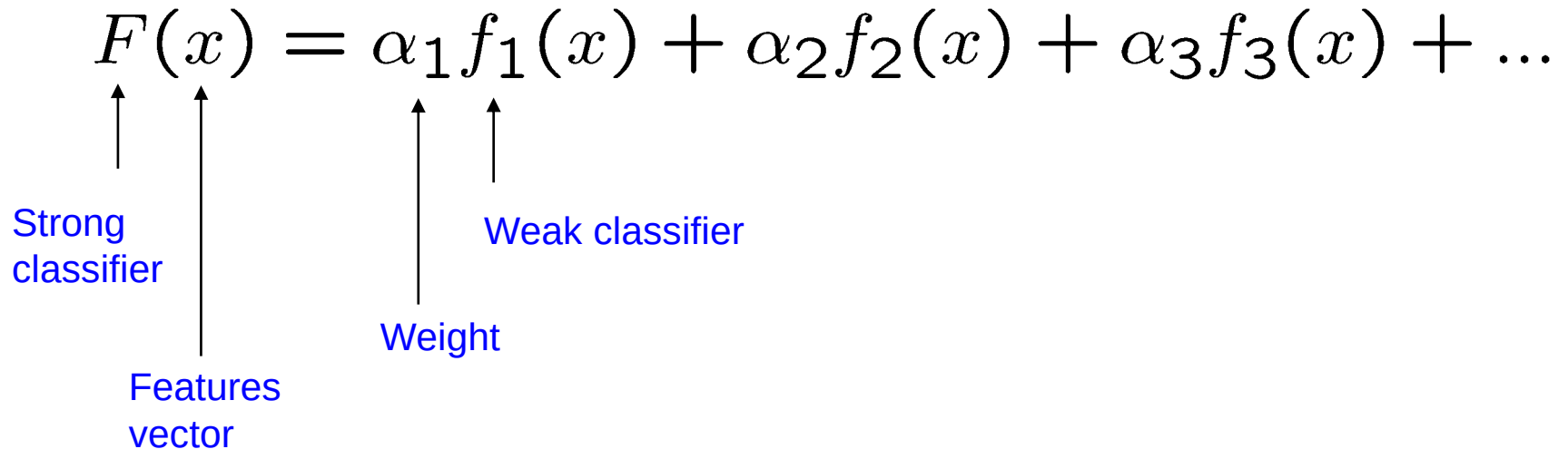
$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) + \dots$$


Diagram illustrating the components of the additive model equation:

- $F(x)$: Strong classifier
- x : Features vector
- α_1 : Weight
- $f_1(x)$: Weak classifier

- We need to define a family of weak classifiers

$f_k(x)$ from a family of weak classifiers

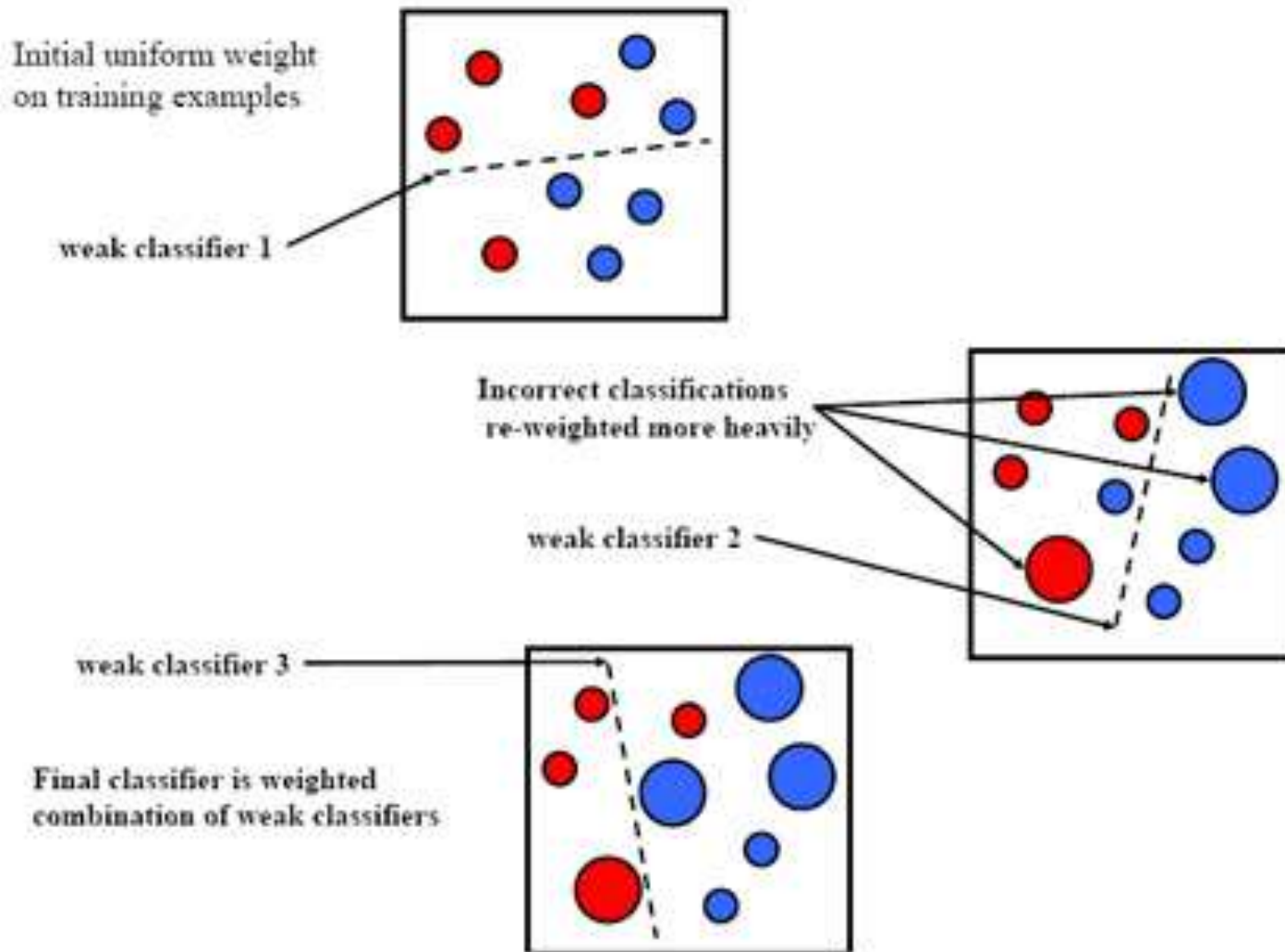
AdaBoost

- ❑ Associate a weight to any samples of the training set
- ❑ For each training cycle a **new** weak learner is provided by an **oracle**
- ❑ The error of the weak learner is the sum of misclassified samples weights
- ❑ According to this error, a weight-vote is associated to the weak learn
 - Low error – high vote
 - High error – low vote

AdaBoost

- AdaBoost changes the weight of the samples after each training stage, increasing the weight of the misclassified samples and reducing the weight of the correct classified:
 - The most misclassified samples have greater weight during training!

AdaBoost: Example

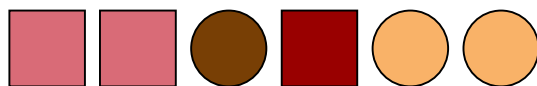


AdaBoost: Example

Training Set 

$D_0(i)$ 0.16 0.16 0.16 0.16 0.16 0.16

1



$$e_1 = 0.16 + 0.16 = 0.32 \ (e < 0.5)$$

$$w_1 = 0.69$$

$D_1(i)$ 0.12 0.12 0.25 0.25 0.12 0.12

2

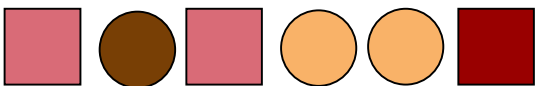


$$e_2 = 0.25 + 0.12 = 0.37 \ (e < 0.5)$$

$$w_2 = 0.51$$

$D_2(i)$ 0.1 0.1 0.2 0.33 0.1 0.16

3



$$e_3 = 0.16 + 0.1 = 0.26 \ (e < 0.5)$$

$$w_3 = 0.73$$

AdaBoost: Example

□ Voting by Majority

Samples



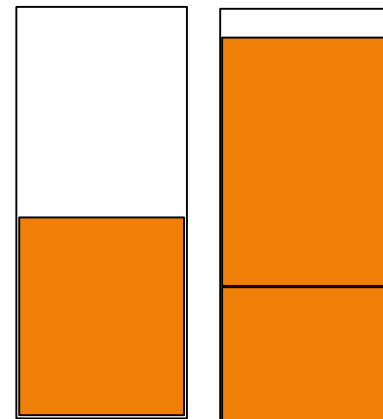
h_1 $w_1=0.69$



h_2 $w_1=0.51$



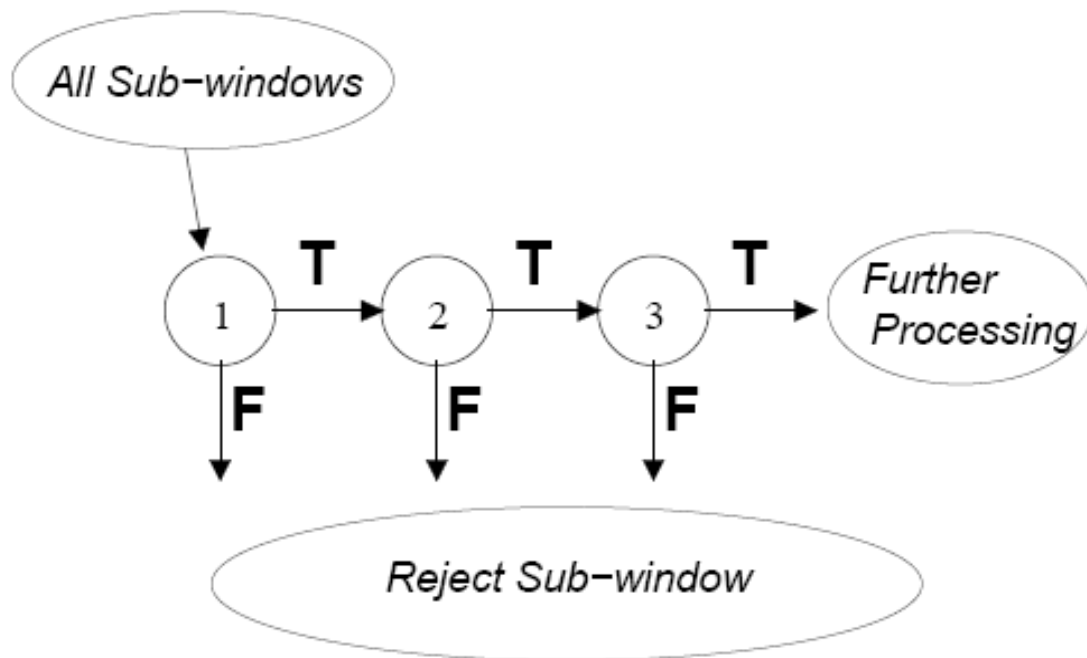
h_3 $w_1=0.73$



Performance of AdaBoost

- ❑ In most practical cases, the ensemble error decreases very rapidly in the first few iterations, and approaches zero or stabilizes as new classifiers are added
- ❑ AdaBoost does not seem to be affected by overfitting; explained by margin theory

A Cascade of Classifiers



❑ 2 approaches:

- Hard Cascade (Viola & Jones): training
- Soft Cascade: post processing

Features

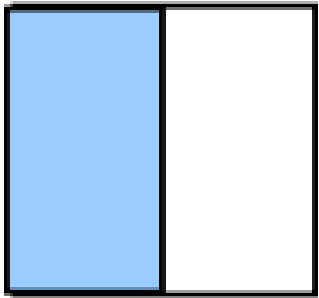
- ❑ Haar Features
- ❑ Histogram of Oriented Gradient
- ❑ Integral Channel Features

Features di Haar

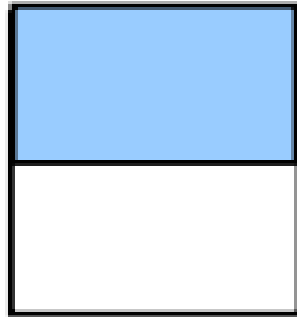


$$S = \sum_{\Omega_B} I - \sum_{\Omega_N} I$$

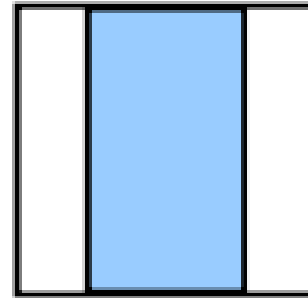
Some examples of Haar features



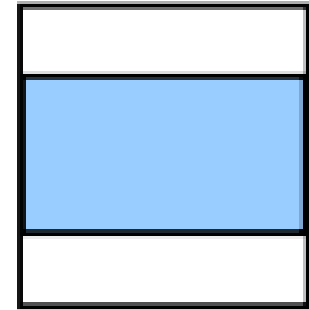
x_2



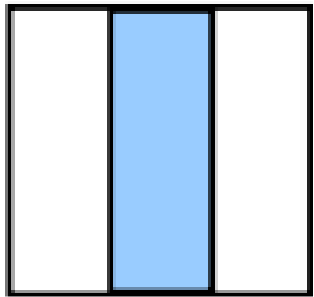
y_2



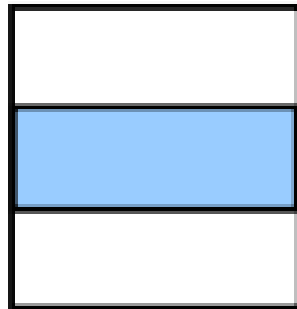
x_4



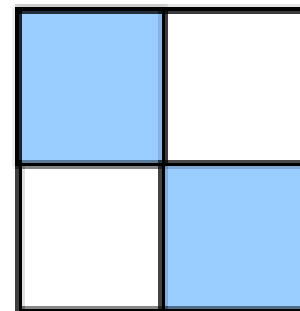
y_4



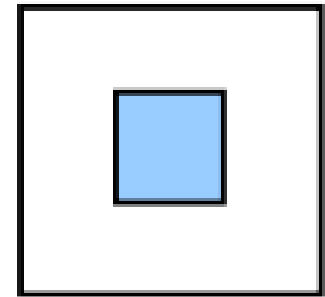
x_3



y_3

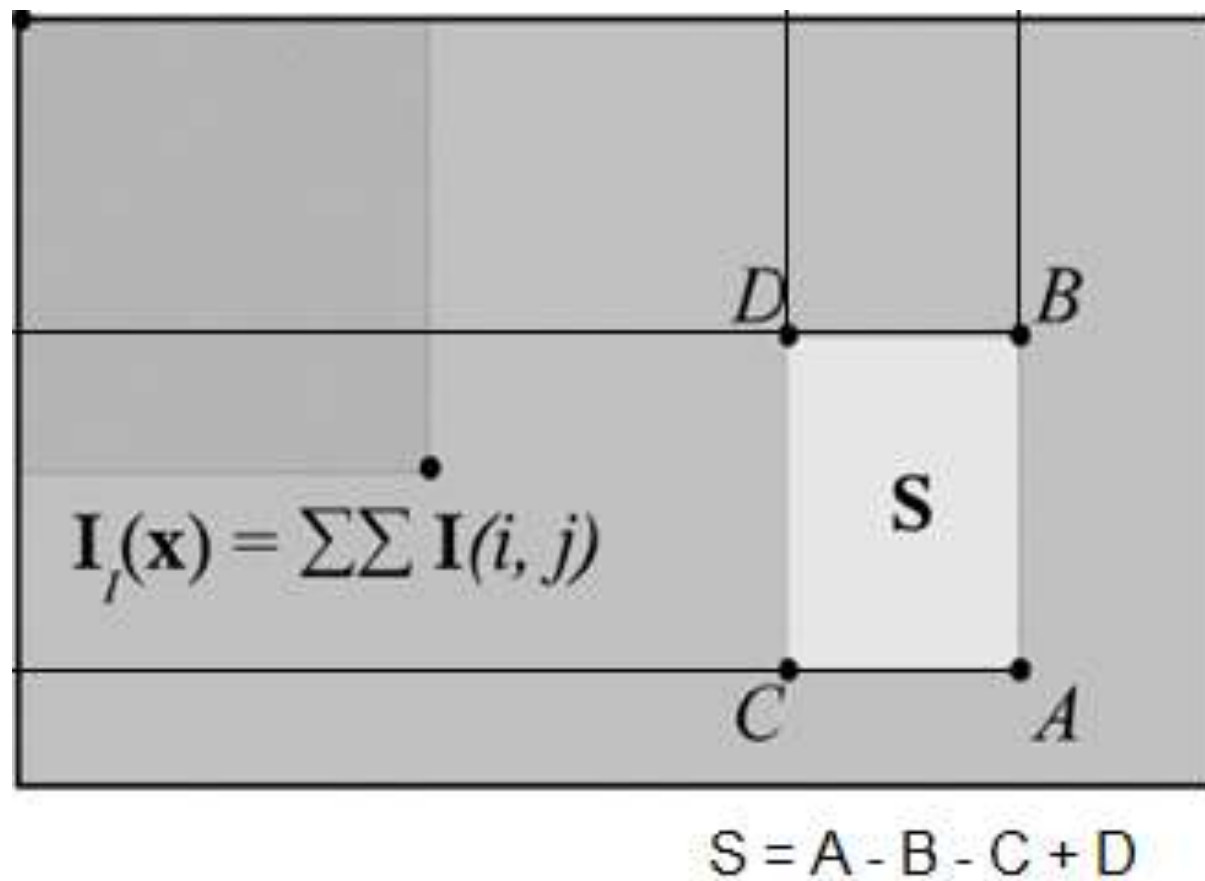


x_2y_2

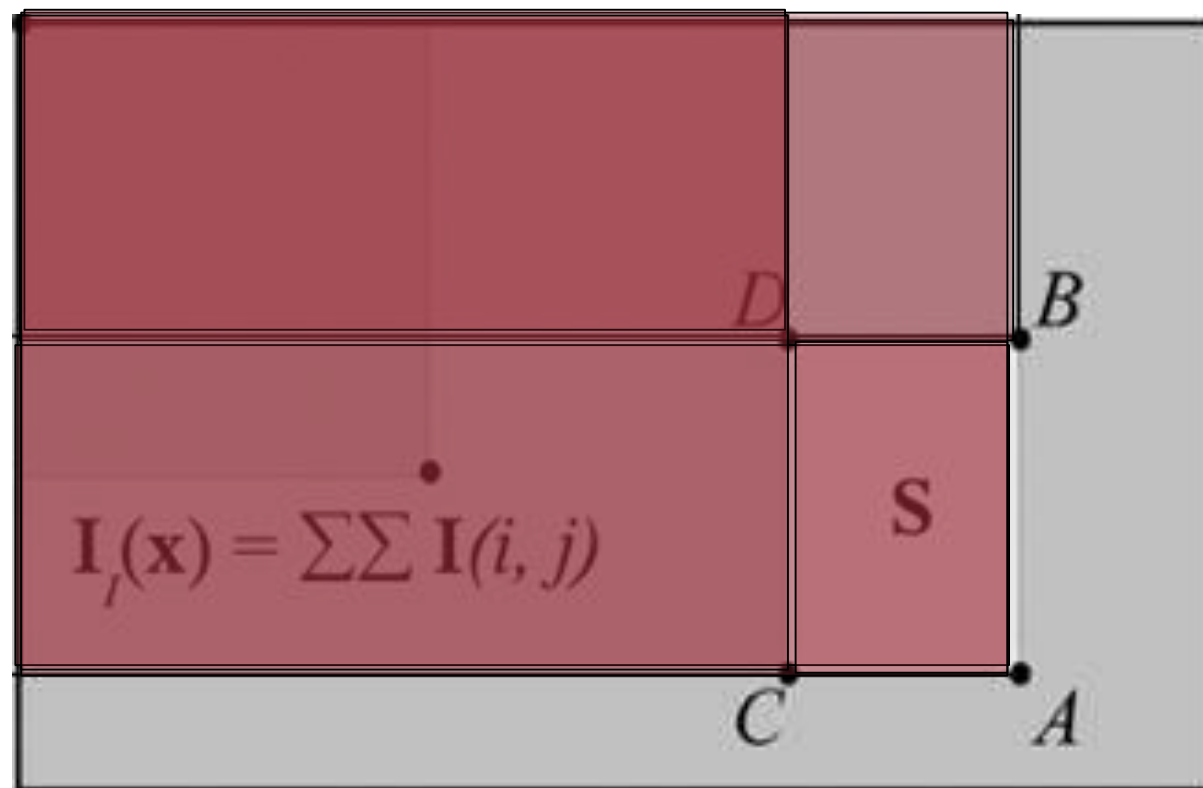


x_3y_3

Integral Image

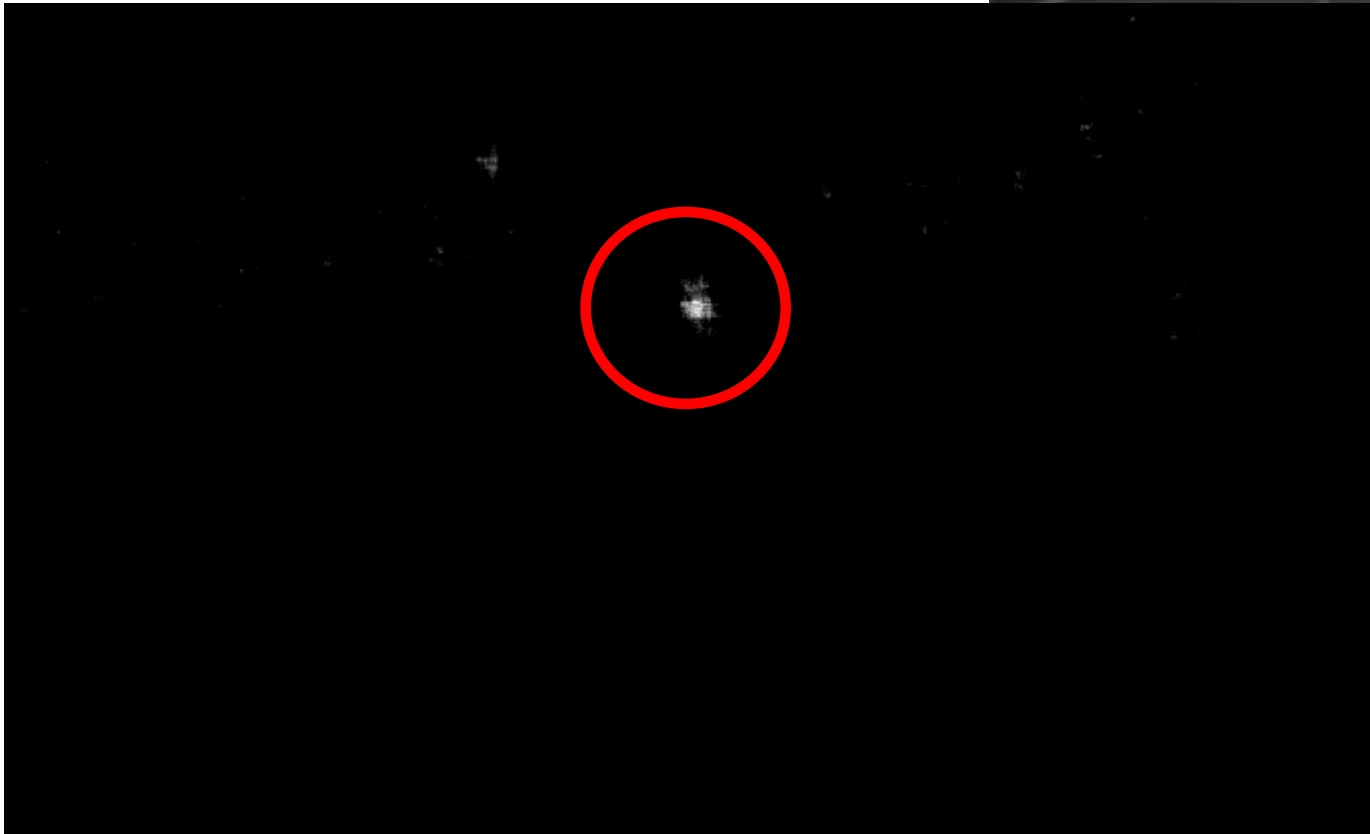


Integral Image

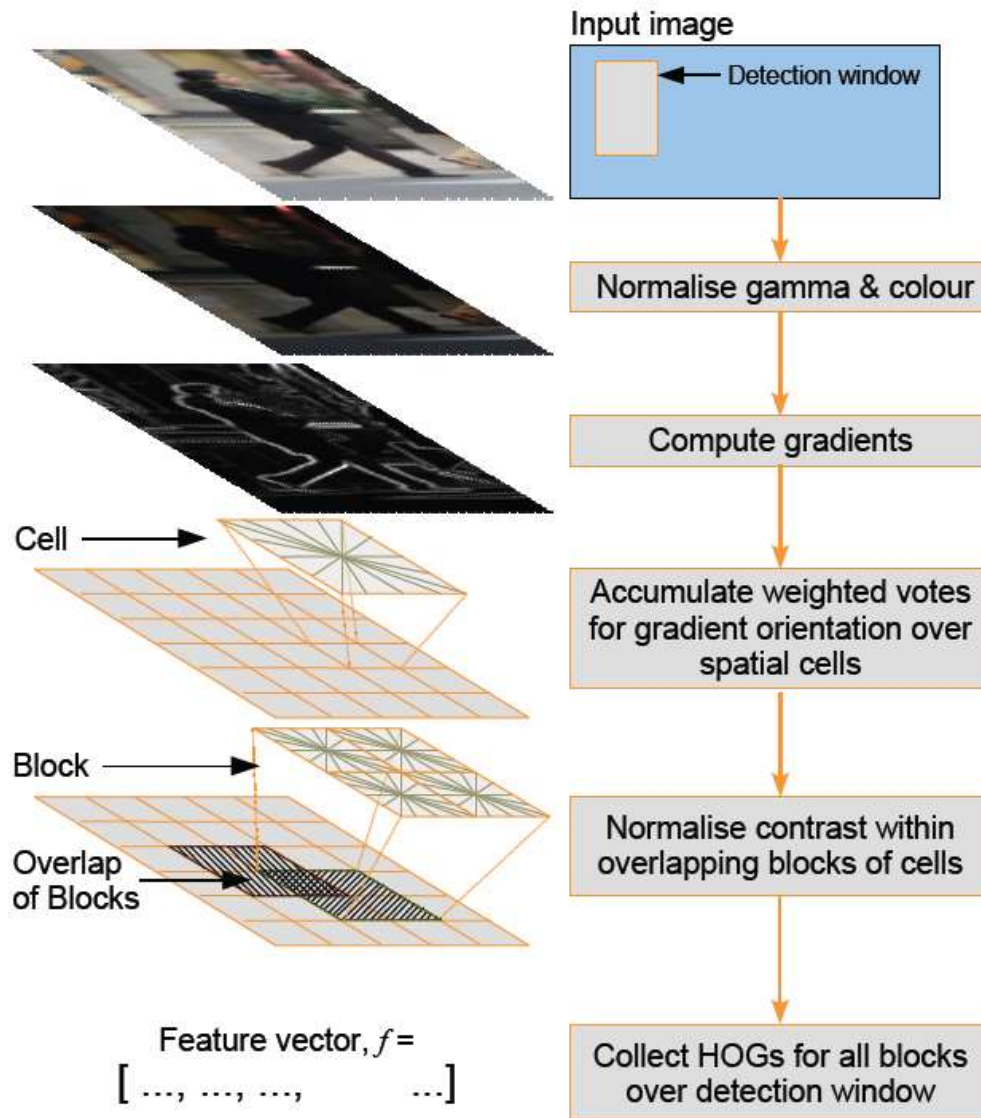


$$S = A - B - C + D$$

Example: Haar and AdaBoost



Histogram of Oriented Gradient



HOG feature vector for one block



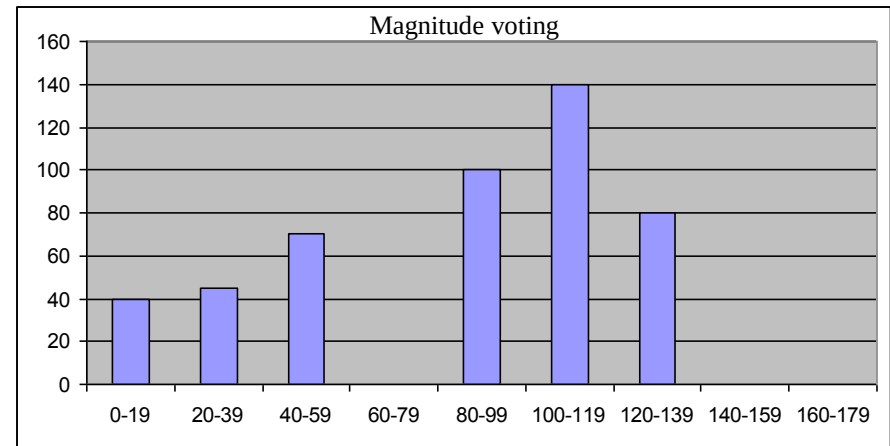
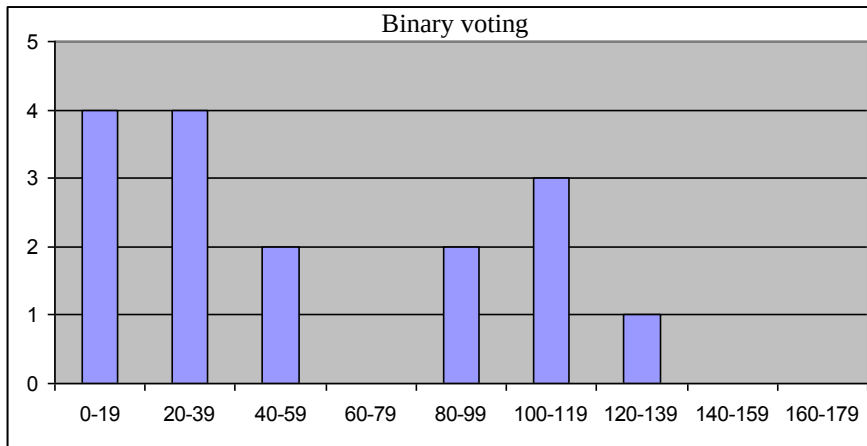
$$f = (h_1^1, \dots, h_9^1, \boxed{h_1^2, \dots, h_9^2}, h_1^3, \dots, h_9^3, h_1^4, \dots, h_9^4)$$

Angle

0	15	25	25
10	15	25	30
45	95	101	110
47	97	101	120

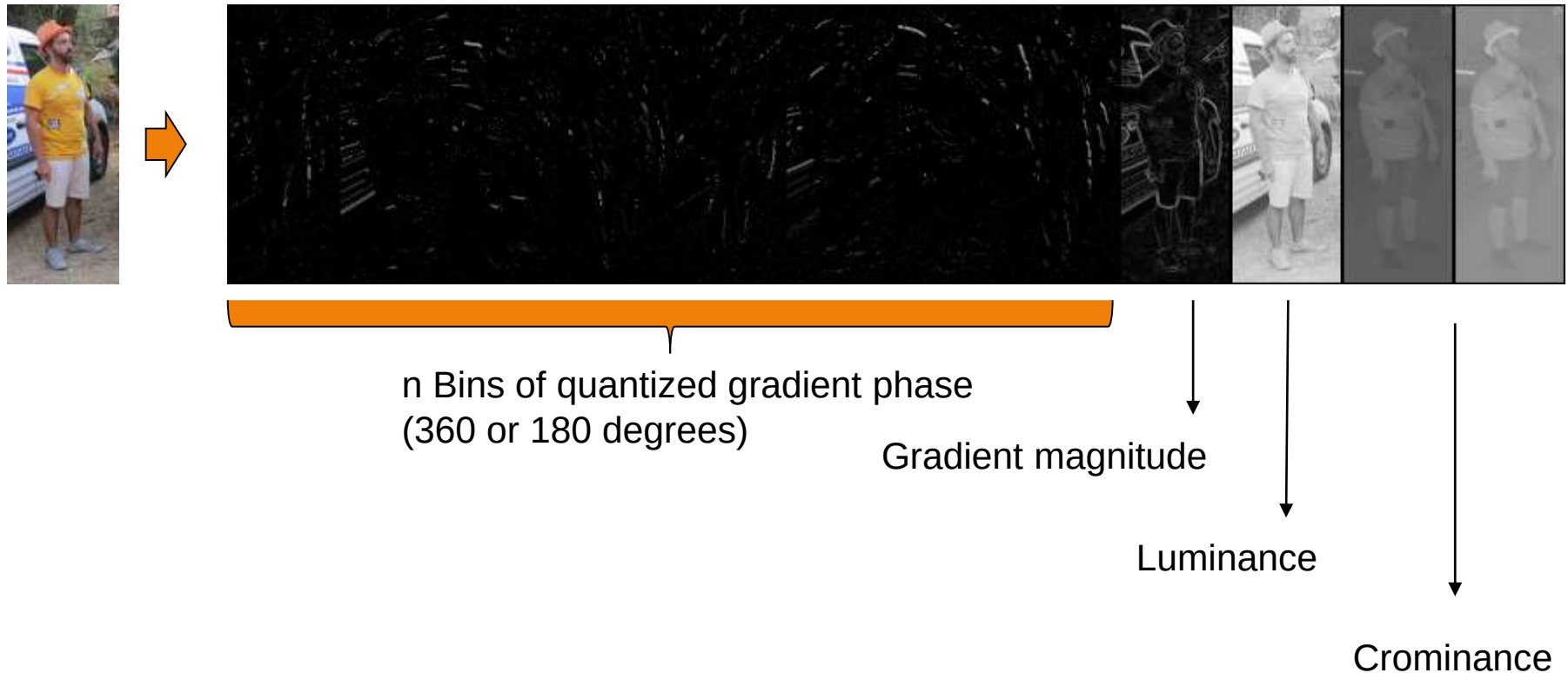
Magnitude

5	20	20	10
5	10	10	5
20	30	30	40
50	70	70	80



Feature vector extends while window moves

Integral Channel Features

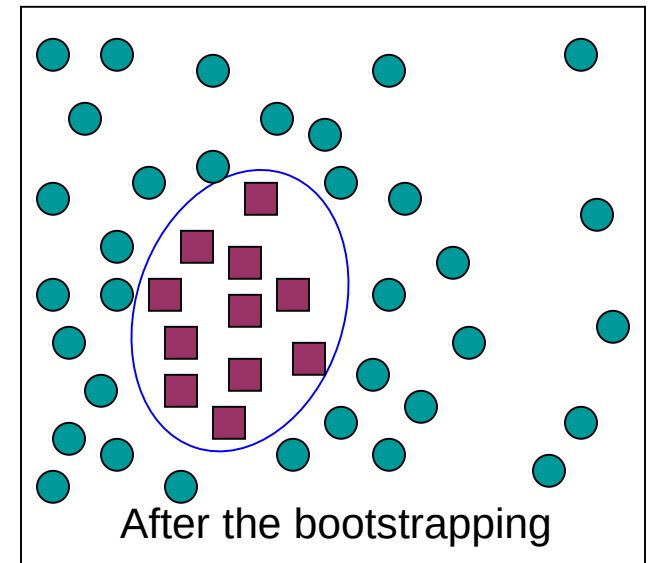
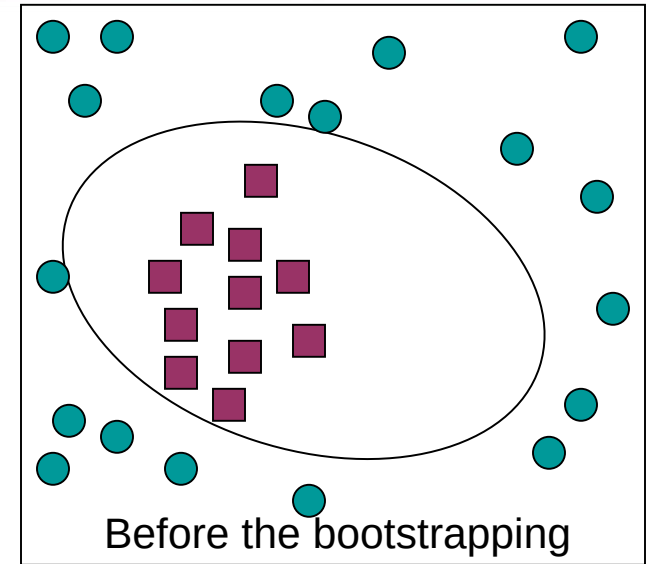


Classifier and friends

- ❑ Effective training: selecting negatives among billions
- ❑ Effective classifying: multiscale, non-maxima suppression

Training the Classifier: Bootstrapping

- ❑ To Obtain a better representation of the negative class in the boundary with the positive class.
- ❑ Procedure:
 1. To train the classifier.
 2. To collect new negative windows that were wrongly classified as positive by the recently trained classifier.
 3. To add these windows to the training set.
 4. Go to (1) if one of the following conditions is not achieved:
 - The false positive rate is not low enough.
 - There is no improvement in the performance of the classifier.



[Sung and Poggio 96]

Non-maximal suppression (NMS)

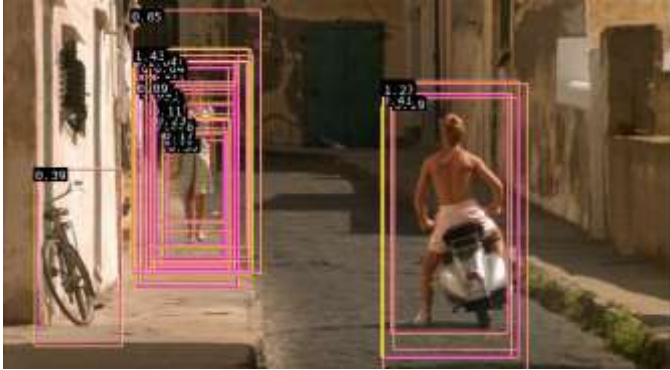


Many detections above threshold.

Non-maximal suppression (NMS)

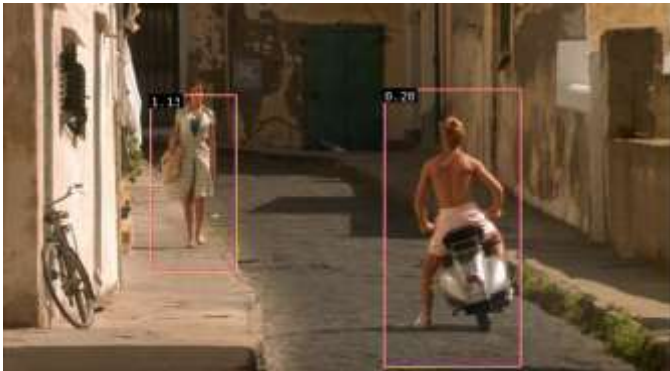
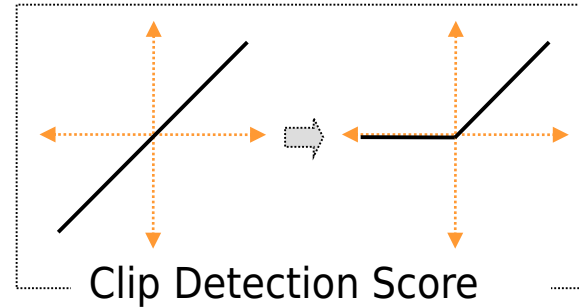


Multi-Scale Object Localisation



Multi-scale dense scan of detection window

Bias



Final detections

Threshold



Apply robust mode detection, like mean shift

- Robust non-maximum suppression is important
- Fine scale transitions helps!

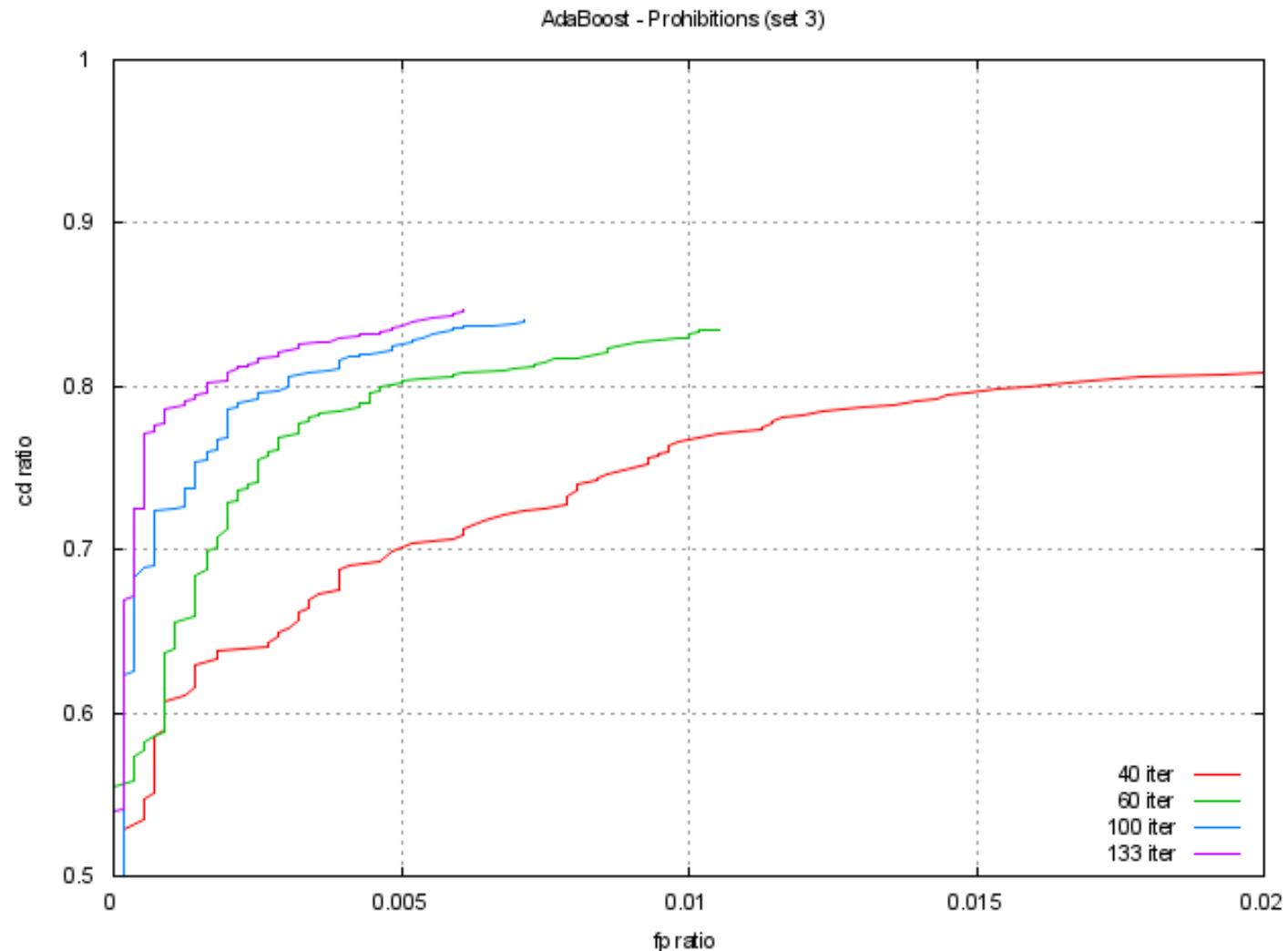
Evaluating Performances

		CLASSIFIER RESPONSE	
		POSITIVE	NEGATIVE
CORRECT RESULT	POSITIVE	TP	FN
	NEGATIVE	FP	TN

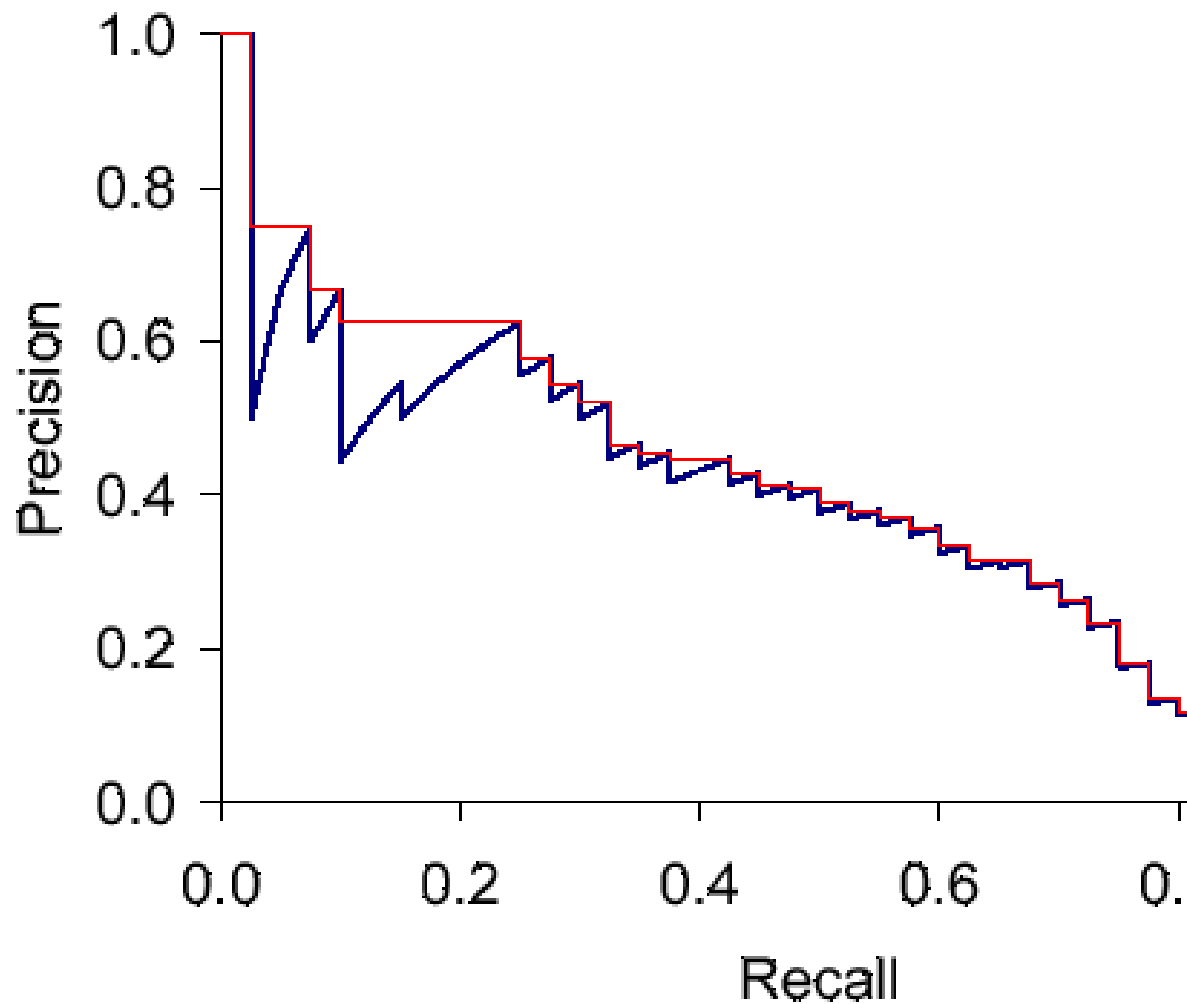
Evaluating Performances

- ❑ True Positive Rate / Recall / Hit-rate:
 $TP / (TP+FN)$
- ❑ Miss rate:
 $1 - TPR$
- ❑ Precision:
 $TP / (TP+FP)$
- ❑ Accuracy:
 $(TP+TN)/(TP+TN+FN+FP)$
- ❑ False Positive (per window, per frame, per minute, per event... etc etc)

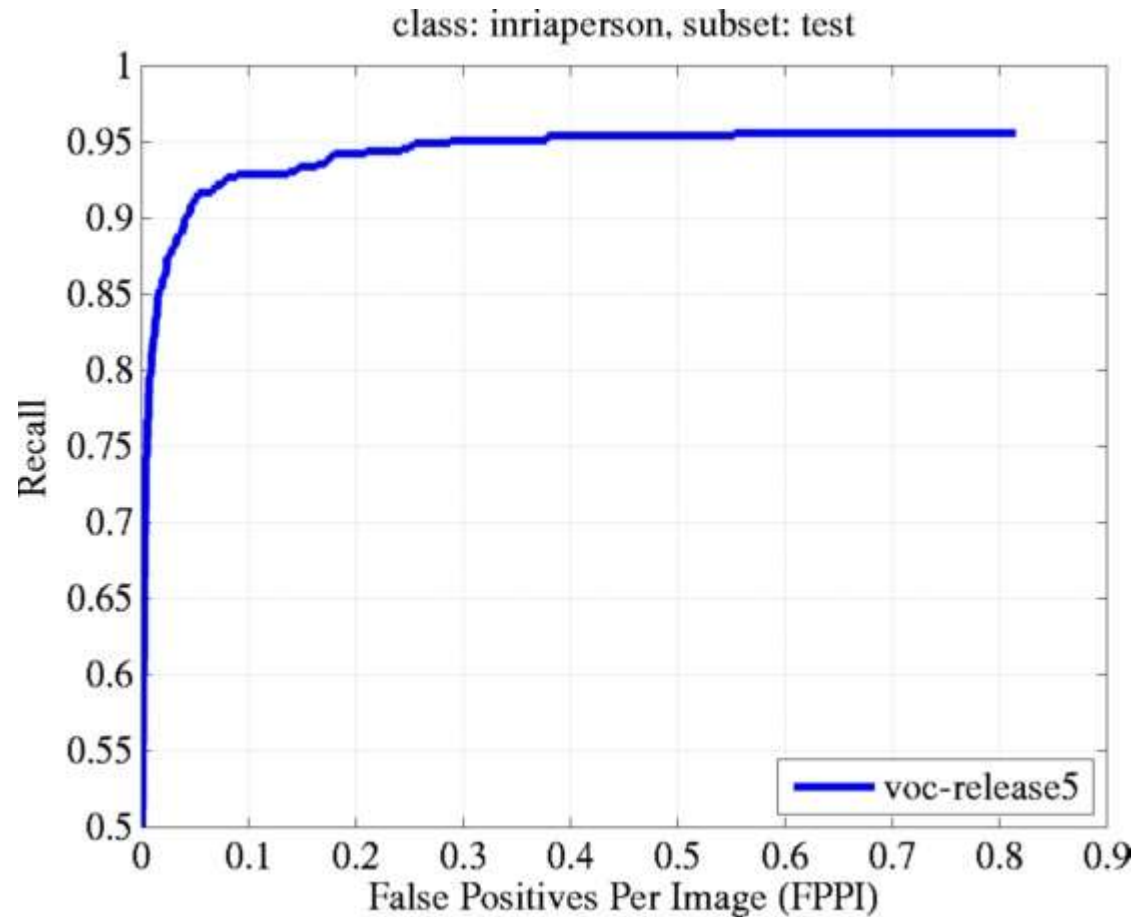
Receiver operating characteristic (ROC)



Precision-Recall curves (PRC)



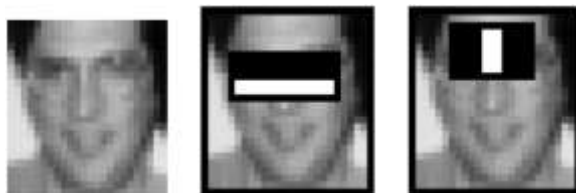
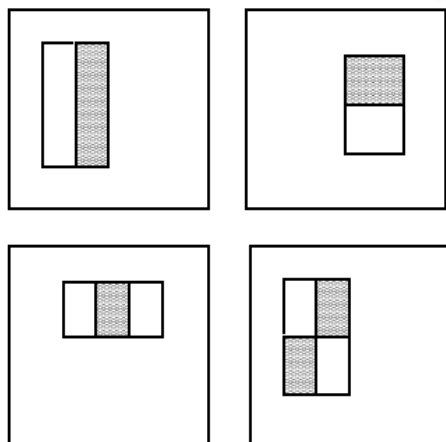
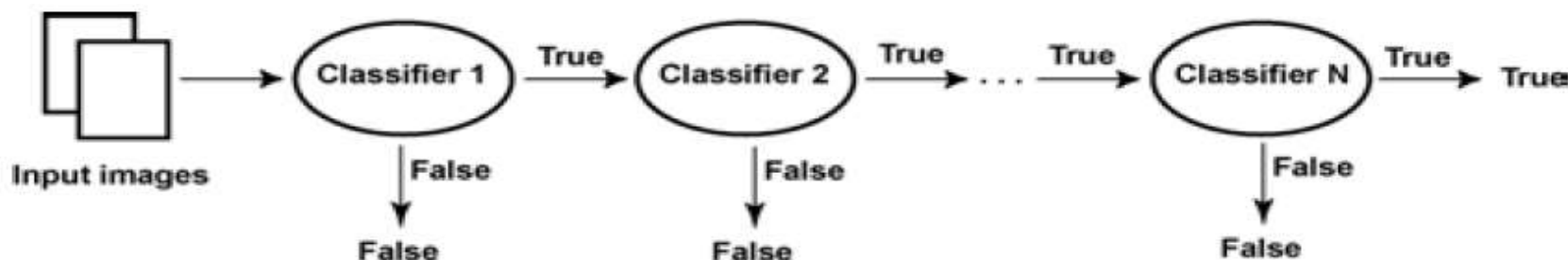
Recall – False Positive rate



AdaBoost Cascade Face Detector

- ❑ A computationally efficient architecture that rapidly rejects unpromising windows
 - A chain of classifiers that each reject some fraction of the negative training samples while keeping almost all positive ones
- ❑ Each classifier is an AdaBoost ensemble of rectangular Haar-like features sampled from a large pool

[Viola & Jones, 2001]

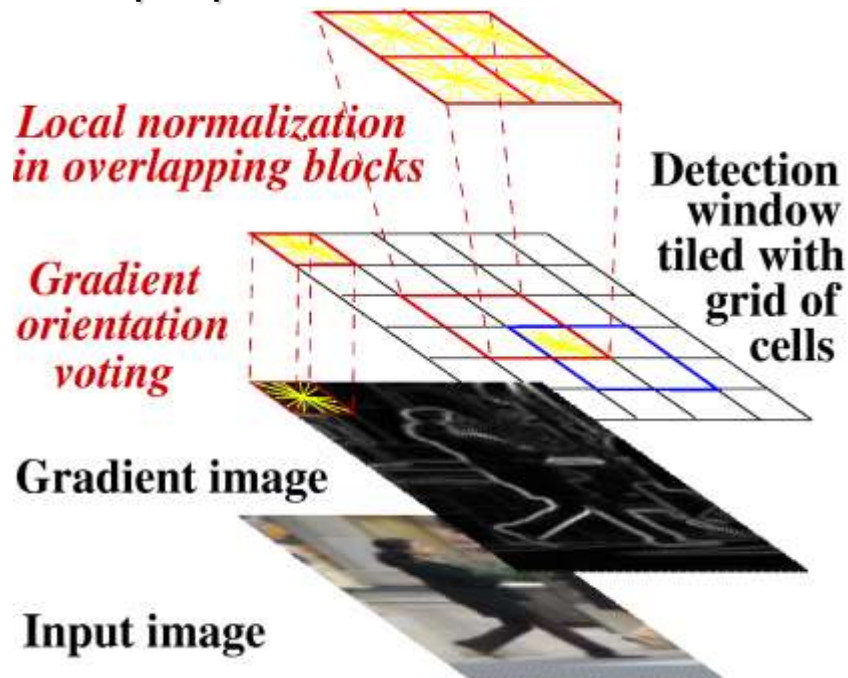


Rectangular Haar features
and the first two features
chosen by AdaBoost

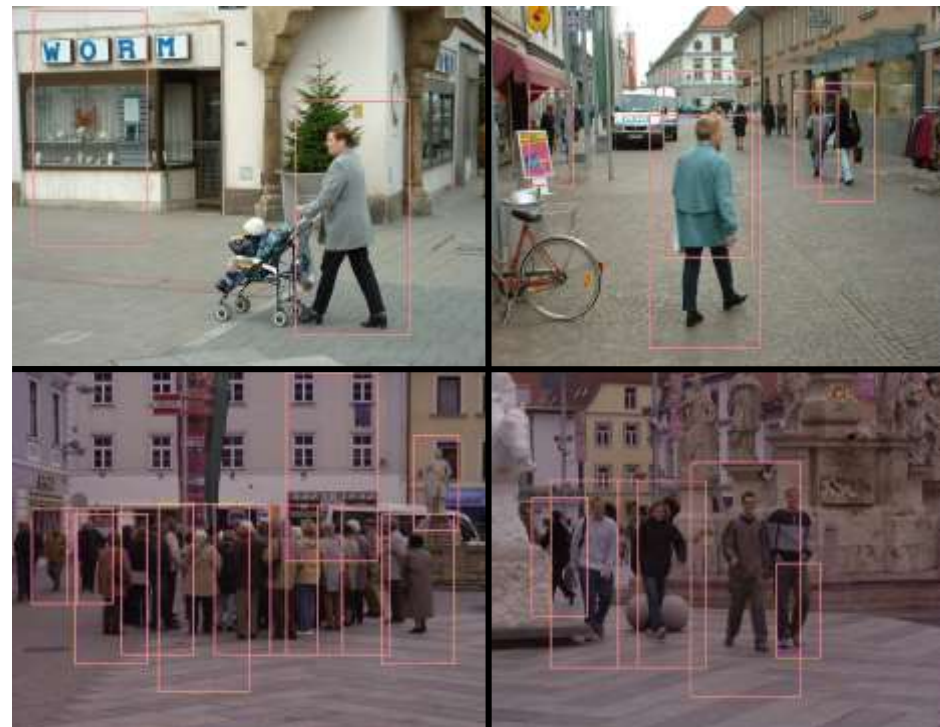
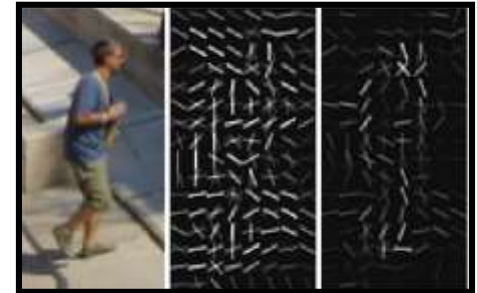


Histogram of Oriented Gradient Human Detector

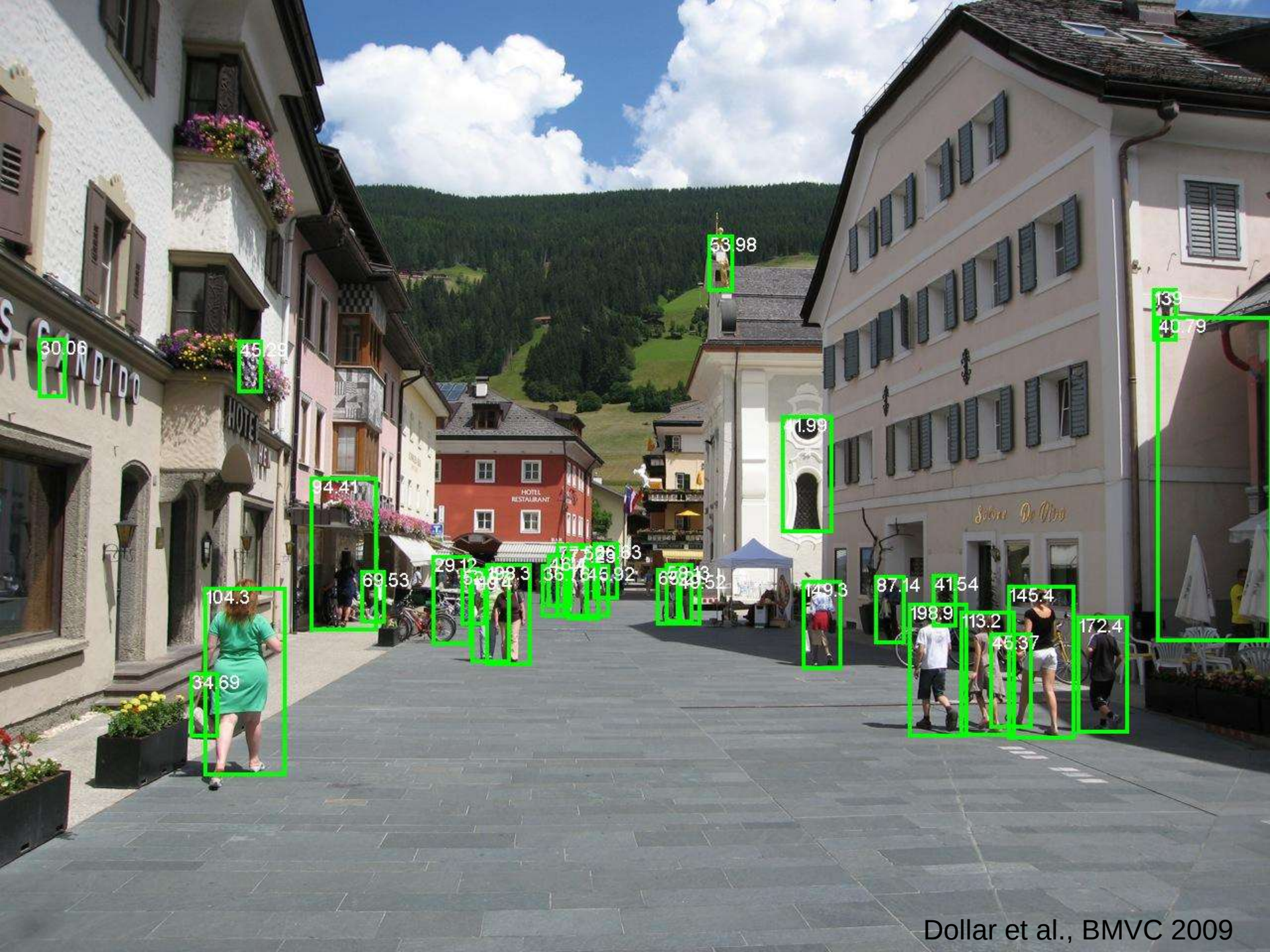
- ❑ Descriptors are a grid of local Histograms of Oriented Gradients (HOG)
- ❑ Linear SVM for runtime efficiency
- ❑ Tolerates different poses, clothing, lighting and background
- ❑ Assumes upright fully visible people



Importance weighted responses



[Dalal & Triggs, CVPR 2005]



30.06

45.29

53.98

139

40.79

94.41

1.99

104.3

69.53

29.12

50.93

46.7

77.53

29.83

36.76

45.92

69.13

69.52

34.69

149.3

87.14

41.54

145.4

198.9

113.2

45.37

172.4

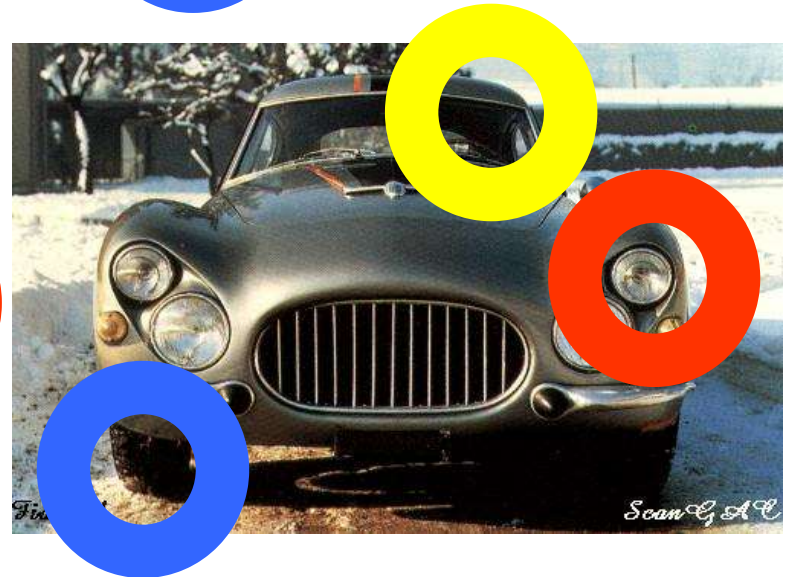
Still a ways to go...



And the Future?

- ❑ New features (ACF?)
- ❑ Convolutional Neural Network ?
- ❑ Part/Patch based model

Representing categories: Parts and Structure



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

Thank you for attention

