Computer Vision – Lecture 7

Sliding-Window based Object Detection

13.05.2019

Bastian Leibe

Visual Computing Institute
RWTH Aachen University
http://www.vision.rwth-aachen.de/

leibe@vision.rwth-aachen.de

RWTH AACHEN UNIVERSITY

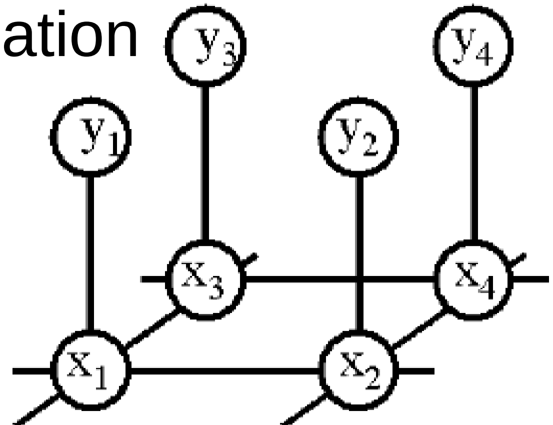
Course Outline

- Image Processing Basics
- Segmentation
  - Segmentation and Grouping
  - Segmentation as Energy Minimization
- Recognition & Categorization
  - Sliding-Window Object Detection
- Local Features & Matching
- Deep Learning
- 3D Reconstruction

Recap: MRFs/CRFs for Image Segmentation

- MRF/CRF formulation

Unary potentials
$\phi(x_i, y_i)$



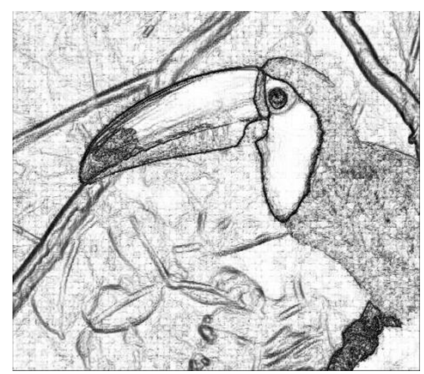
Pairwise potentials
$\psi(x_i, x_j)$

$\Rightarrow$ Minimize the energy

$$E(\mathbf{x}, \mathbf{y}) = \sum_i \phi(x_i, y_i) + \sum_{i,j} \psi(x_i, x_j)$$

Data (D)

Unary likelihood

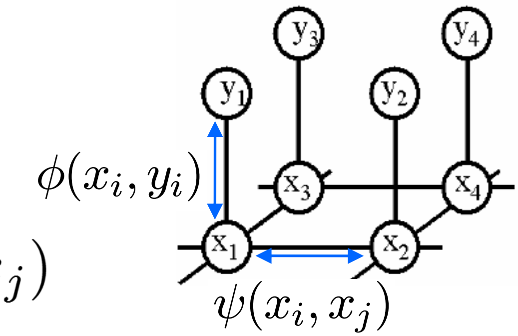


Pair-wise Terms

MAP Solution

Slide adapted from Phil Torr

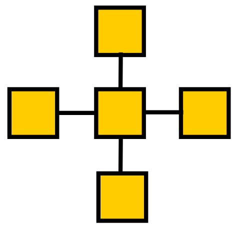
Recap: Energy Formulation



- Energy function

$$E(\mathbf{x}, \mathbf{y}) = \sum_i \underbrace{\phi(x_i, y_i)}_{\text{Unary potentials}} + \sum_{i,j} \underbrace{\psi(x_i, x_j)}_{\text{Pairwise potentials}}$$

- Unary potentials ϕ
  - Encode local information about the given pixel/patch
  - How likely is a pixel/patch to belong to a certain class (e.g. foreground/background)?
- Pairwise potentials ψ
  - Encode neighborhood information
  - How different is a pixel/patch's label from that of its neighbor? (e.g. based on intensity/color/texture difference, edges)

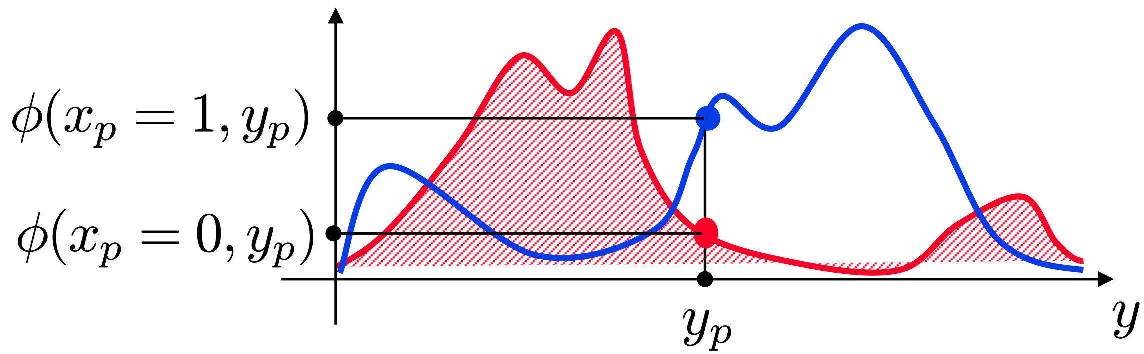


Recap: How to Set the Potentials?

- Unary potentials
  - E.g. color model, modeled with a Mixture of Gaussians

$$\phi(x_i, y_i; \theta_\phi) = \log \sum_k \theta_\phi(x_i, k) p(k|x_i) \mathcal{N}(y_i; \bar{y}_k, \Sigma_k)$$

$\Rightarrow$ Learn color distributions for each label



Recap: How to Set the Potentials?

- Pairwise potentials
  - Potts Model

$$\psi(x_i, x_j; \theta_\psi) = \theta_\psi \delta(x_i \neq x_j)$$

    – Simplest discontinuity preserving model.
    – Discontinuities between any pair of labels are penalized equally.
    – Useful when labels are unordered or number of labels is small.

  - Extension: "Contrast sensitive Potts model"

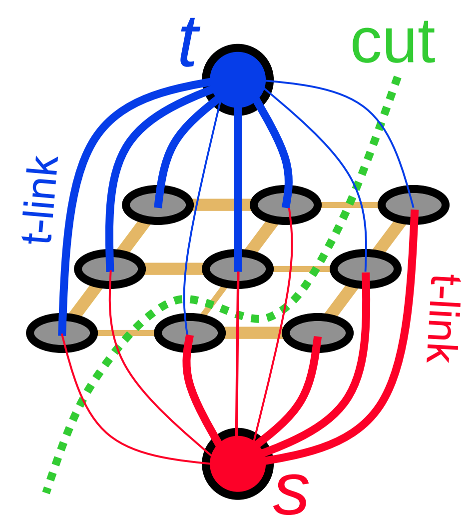
$$\psi(x_i, x_j, g_{ij}(\mathbf{y}); \theta_\psi) = -\theta_\psi g_{ij}(\mathbf{y}) \delta(x_i \neq x_j)$$

where

$$g_{ij}(\mathbf{y}) = e^{-\beta \|y_i - y_j\|^2} \qquad \beta = \frac{1}{2}\left(\mathrm{avg}\left(\|y_i - y_j\|^2\right)\right)^{-1}$$

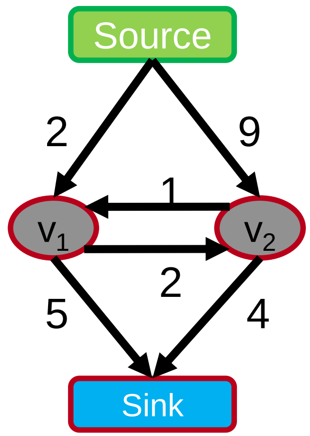
$\Rightarrow$ Discourages label changes except in places where there is also a large change in the observations.

Recap: Graph-Cuts Energy Minimization



- Solve an equivalent graph cut problem
  1. Introduce extra nodes: source and sink
  2. Weight connections to source/sink (t-links) by $\phi(x_i = s)$ and $\phi(x_i = t)$, respectively.
  3. Weight connections between nodes (n-links) by $\psi(x_i, x_j)$.
  4. Find the minimum cost cut that separates source from sink.

  $\Rightarrow$ Solution is equivalent to minimum of the energy.



- s-t Mincut can be solved efficiently
  - Dual to the well-known max flow problem
  - Very efficient algorithms available for regular grid graphs (1-2 MPixels/s)
  - Globally optimal result for 2-class problems

Recap: When Can s-t Graph Cuts Be Applied?

$$E(L) = \underbrace{\sum_{p} E_p(L_p)}_{\text{t-links}}^{\text{Unary potentials}} + \underbrace{\sum_{pq \in N} E(L_p, L_q)}_{\text{n-links}}^{\text{Pairwise potentials}} \qquad L_p \in \{s,t\}$$

- s-t graph cuts can only globally minimize binary energies that are submodular. [Boros & Hummer, 2002, Kolmogorov & Zabih, 2004]

$E(L)$ can be minimized by s-t graph cuts $\Leftrightarrow$ $\boxed{E(s,s)+E(t,t) \le E(s,t)+E(t,s)}$

Submodularity ("convexity")

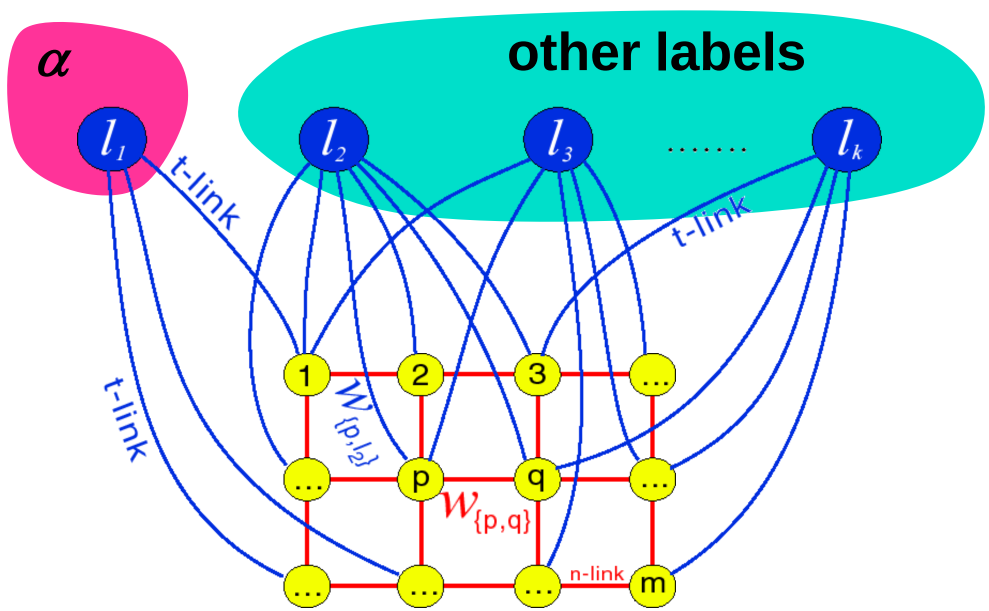
- Submodularity is the discrete equivalent to convexity.
  - Implies that every local energy minimum is a global minimum.
  - $\Rightarrow$ Solution will be globally optimal.

Dealing with Non-Binary Cases

- Limitation to binary energies is often a nuisance.
  $\Rightarrow$ E.g. binary segmentation only…
- We would like to solve also multi-label problems.
  - The bad news: Problem is NP-hard with 3 or more labels!
- There exist some approximation algorithms which extend graph cuts to the multi-label case:
  - α-Expansion
  - $\alpha\beta$-Swap
- They are no longer guaranteed to return the globally optimal result.
  - But α-Expansion has a guaranteed approximation quality (2-approx) and converges in a few iterations.

α-Expansion Move

- Basic idea:
  - Break multi-way cut computation into a sequence of binary s-t cuts.

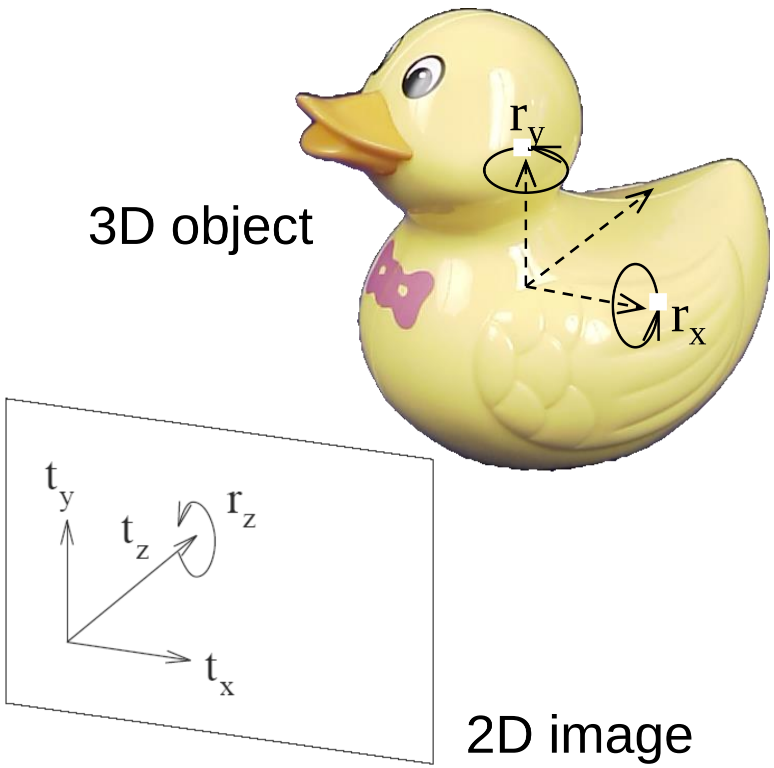


Topics of This Lecture

- Object Recognition and Categorization
  - Problem Definitions
  - Challenges
- Sliding-Window based Object Detection
  - Detection via Classification
  - Global Representations
  - Classifier Construction
- Classification with SVMs
  - Support Vector Machines
  - HOG Detector
- Classification with Boosting
  - AdaBoost
  - Viola-Jones Face Detection

Object Recognition: Challenges

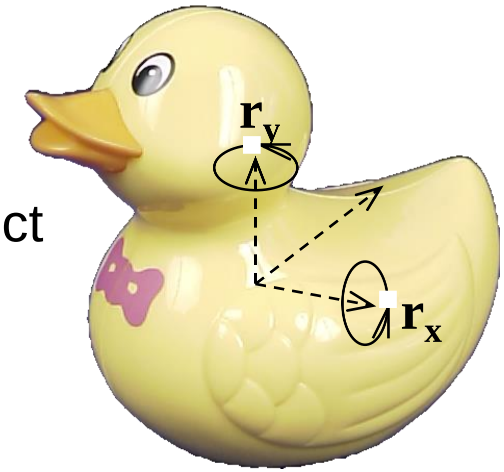
- Viewpoint changes
  - Translation
  - Image-plane rotation
  - Scale changes
  - Out-of-plane rotation
- Illumination
- Noise
- Clutter
- Occlusion



Appearance-Based Recognition

- Basic assumption
  - Objects can be represented by a set of images (“appearances”).
  - For recognition, it is sufficient to just compare the 2D appearances.
  - No 3D model is needed.

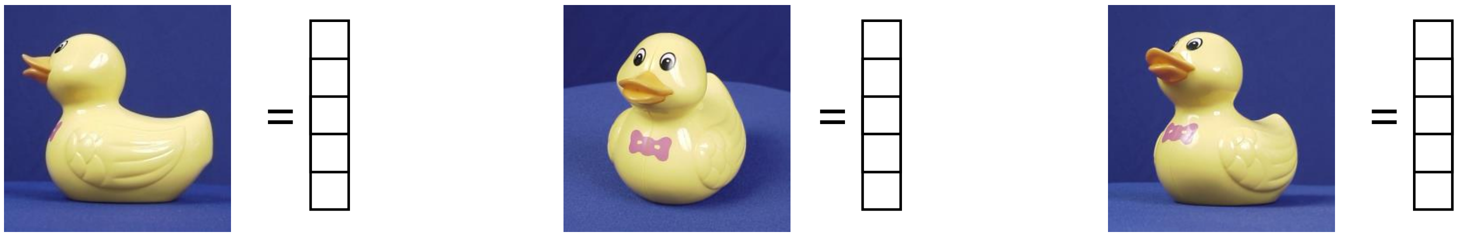
3D object



$\Rightarrow$ Fundamental paradigm shift in the 90’s

Global Representation

- Idea
  - Represent each object (view) by a global descriptor.

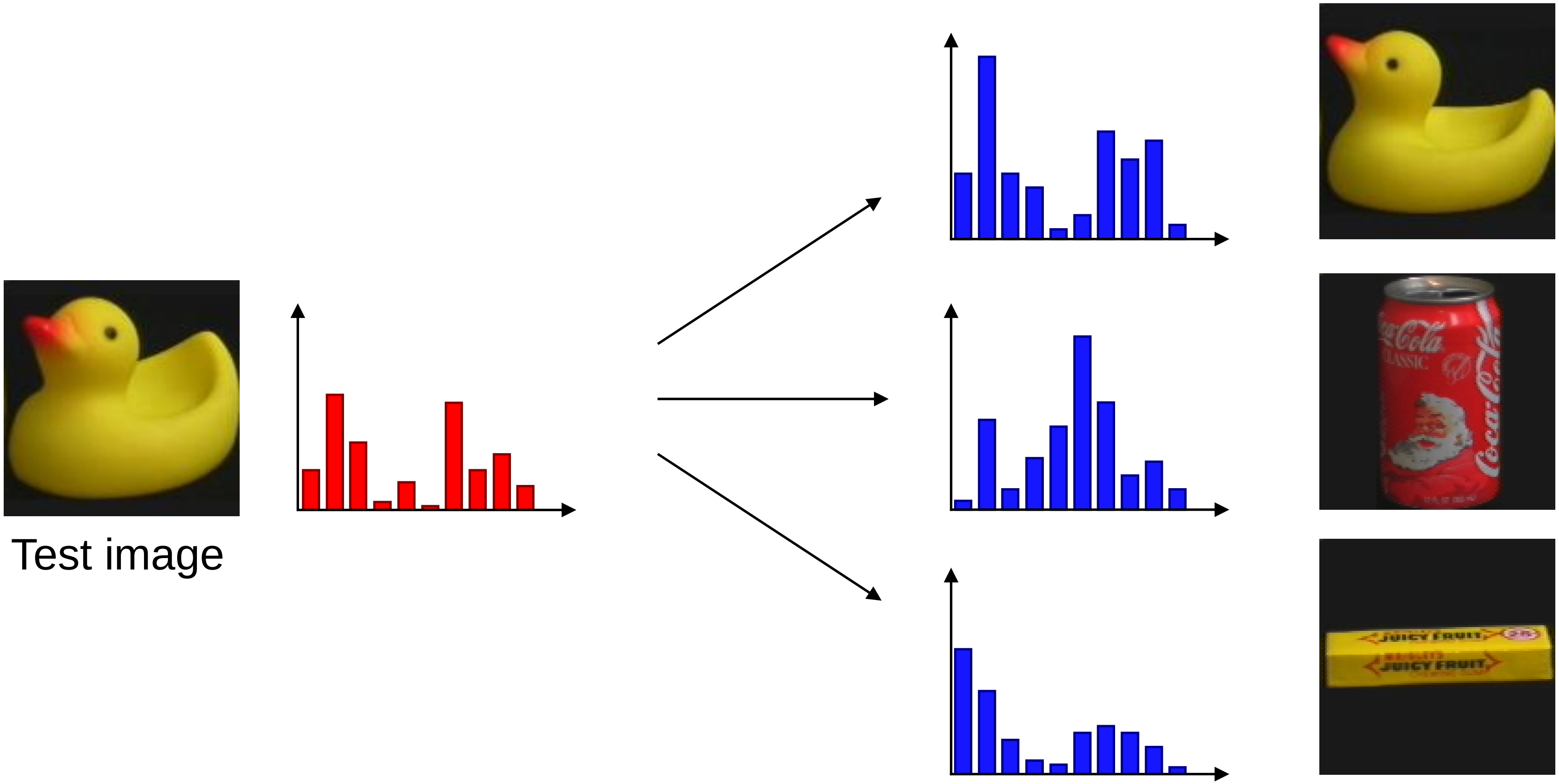


  - For recognizing objects, just match the descriptors.
  - Some modes of variation are built into the descriptor, the others have to be incorporated in the training data.
    - E.g., a descriptor can be made invariant to image-plane rotations.
    - Other variations:

Viewpoint changes	Illumination
– Translation	Noise
– Scale changes	Clutter
– Out-of-plane rotation	Occlusion

Appearance based Recognition

- Recognition as feature vector matching

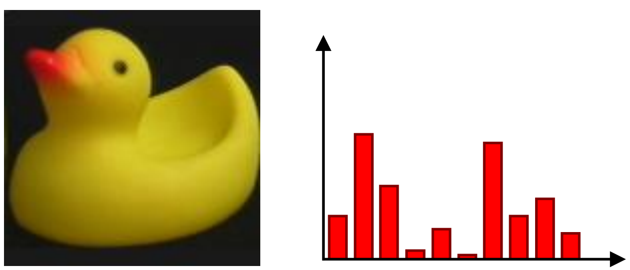


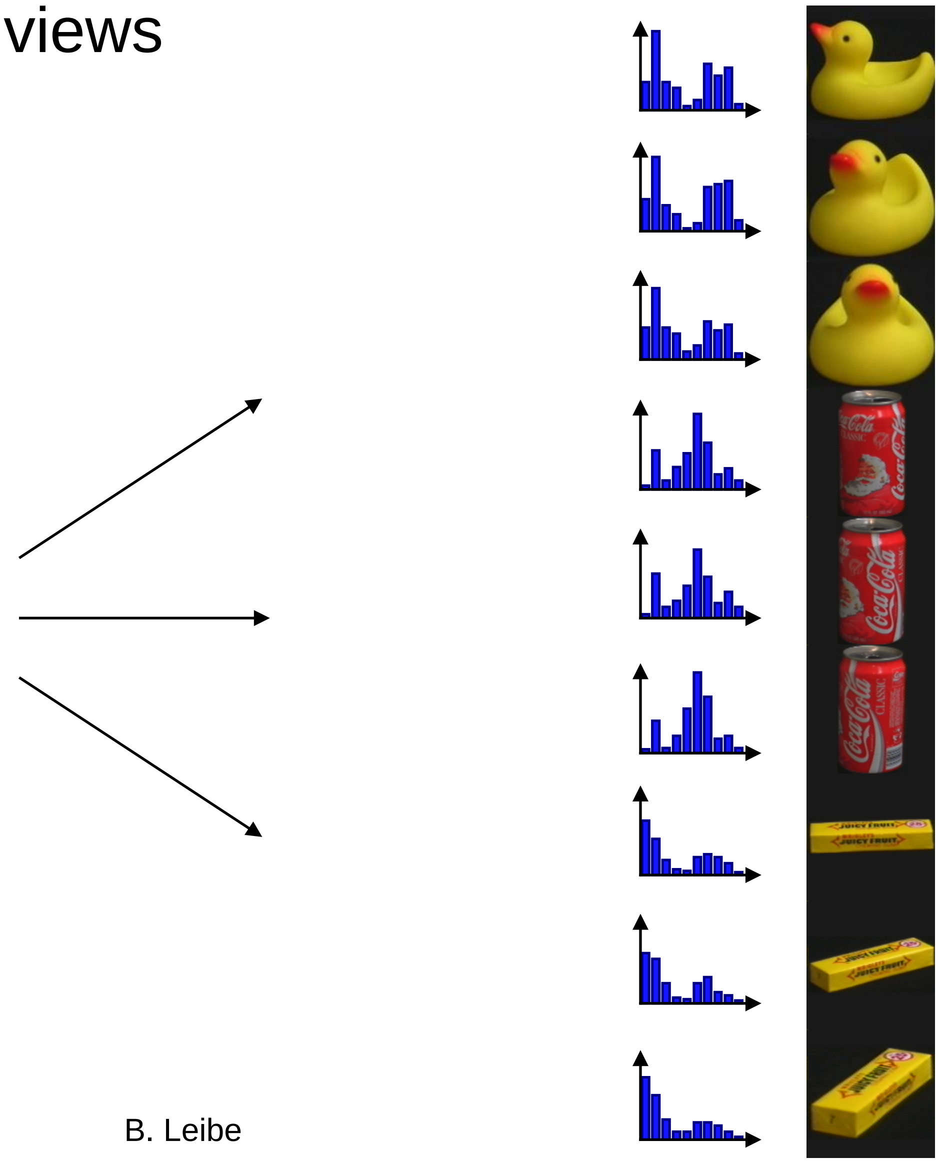
Test image

Known objects

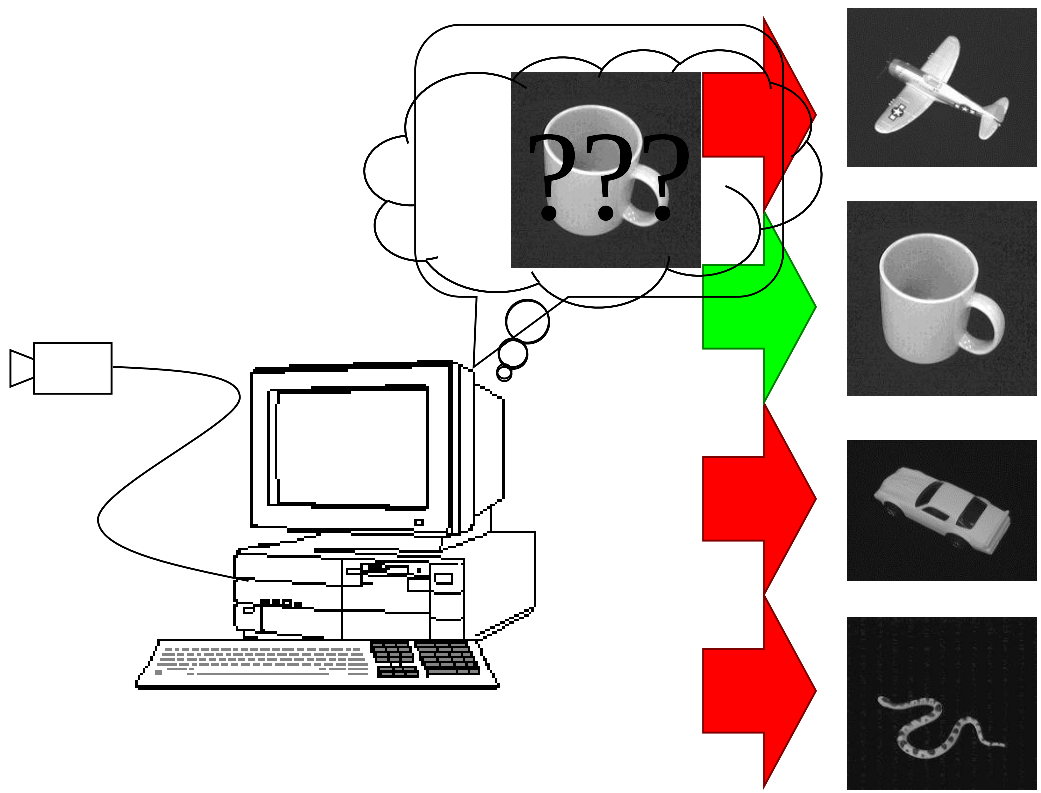
Appearance based Recognition

- With multiple training views


Test image

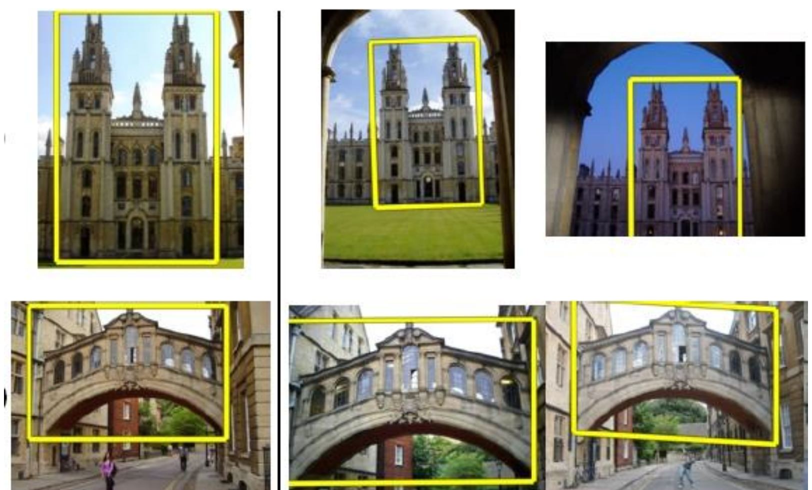


Identification vs. Categorization



Identification vs. Categorization

- Find *this particular* object



- Recognize ANY car

- Recognize ANY cow

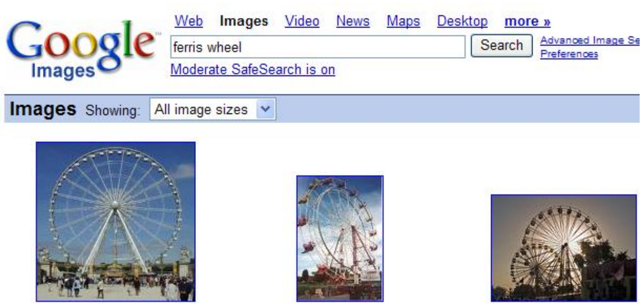
Object Categorization – Potential Applications

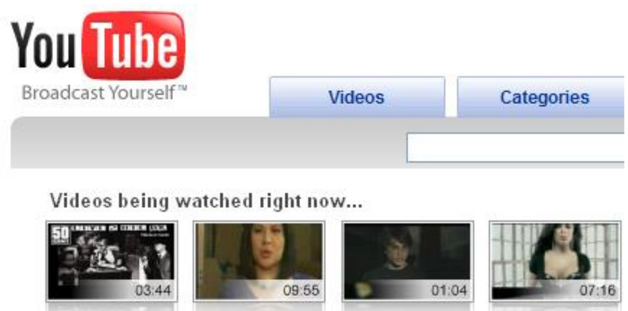
There is a wide range of applications, including…

Autonomous robots

Navigation, driver safety

Consumer electronics





Content-based retrieval and analysis for images and videos

Medical image analysis

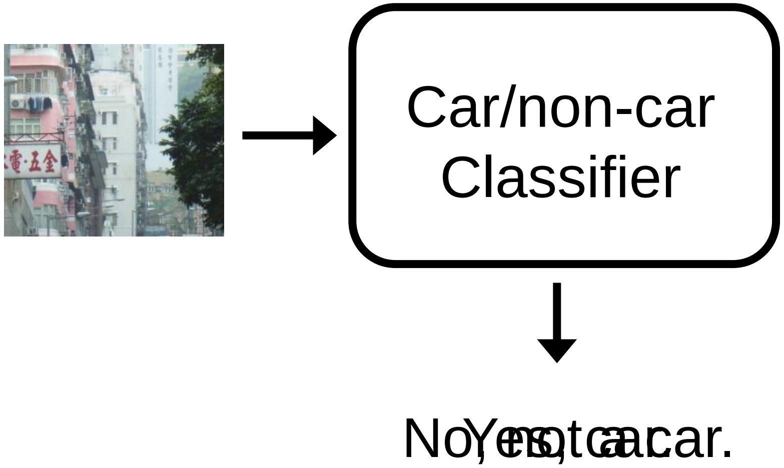
Slide adapted from Kristen Grauman

Topics of This Lecture

- Object Categorization
  - Problem Definition
  - Challenges
- Sliding-Window based Object Detection
  - Detection via Classification
  - Global Representations
  - Classifier Construction
- Classification with SVMs
  - Support Vector Machines
  - HOG Detector
- Classification with Boosting
  - AdaBoost
  - Viola-Jones Face Detection

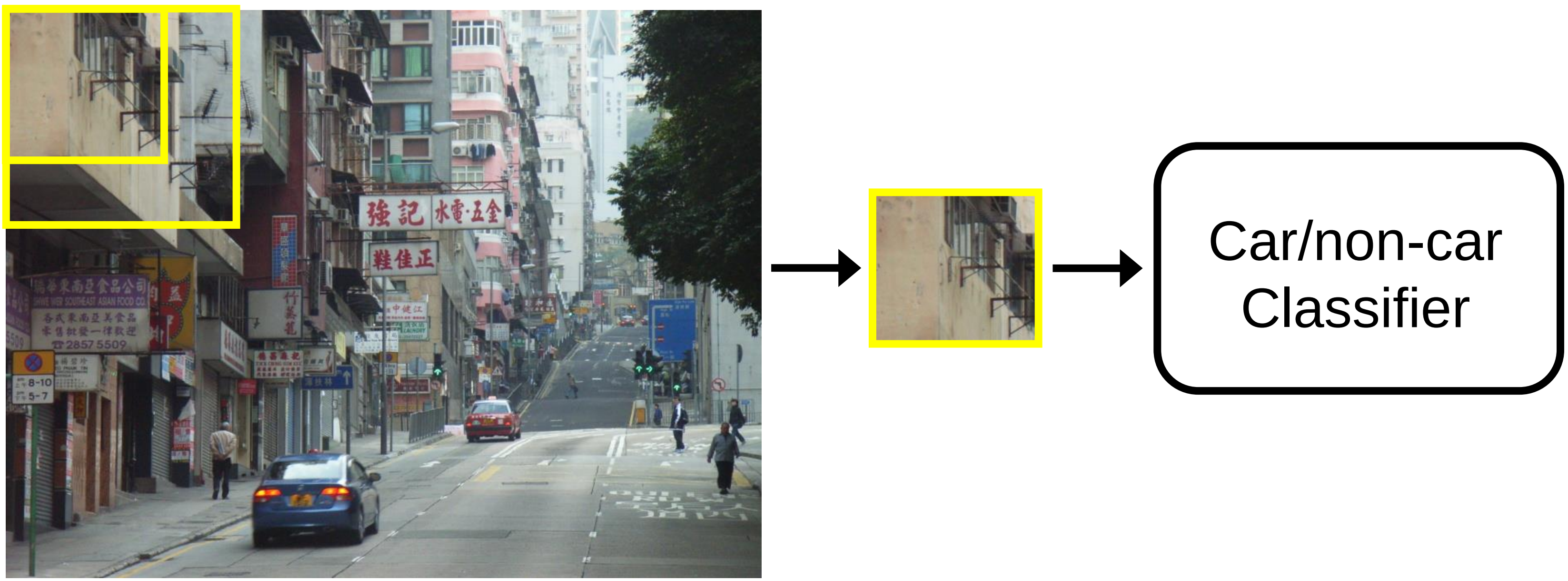
Detection via Classification: Main Idea

- Basic component: a binary classifier



Detection via Classification: Main Idea

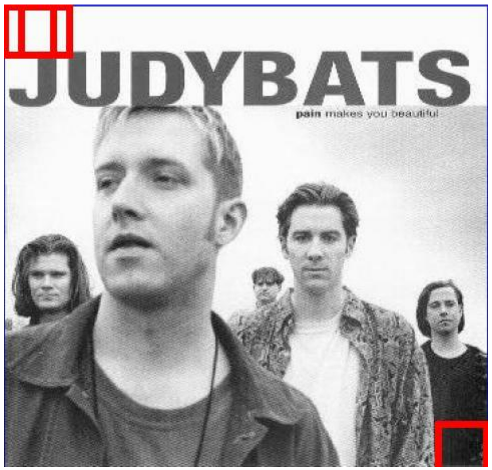
- If the object may be in a cluttered scene, slide a window around looking for it.



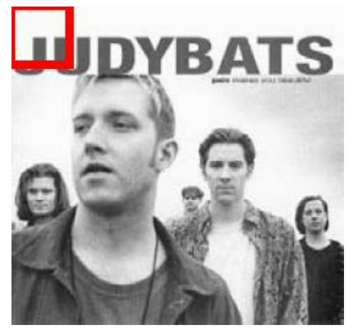
- Essentially, this is a brute-force approach with many local decisions.

What *is* a Sliding Window Approach?

- Search over space and scale



...



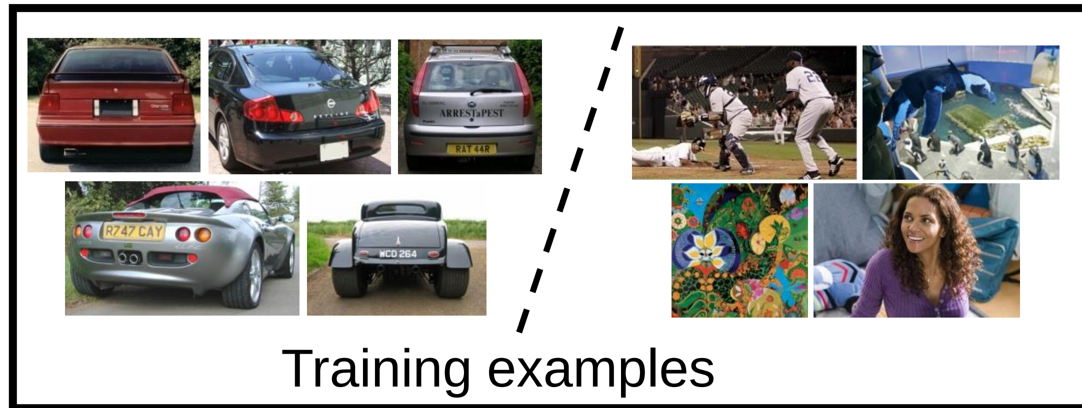
...

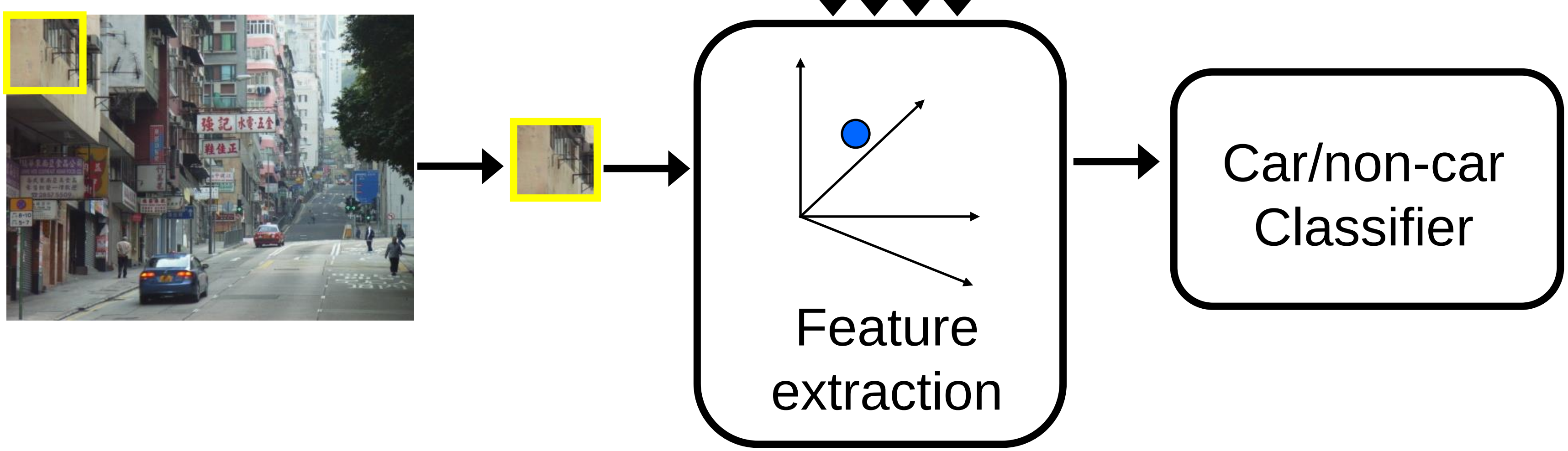
- Detection as subwindow classification problem
- *“In the absence of a more intelligent strategy, any global image classification approach can be converted into a localization approach by using a sliding-window search.”*

Detection via Classification: Main Idea

Fleshing out this pipeline a bit more, we need to:

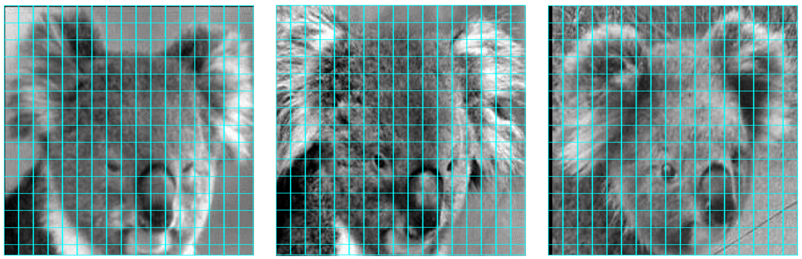
1. Obtain training data
2. Define features
3. Define classifier

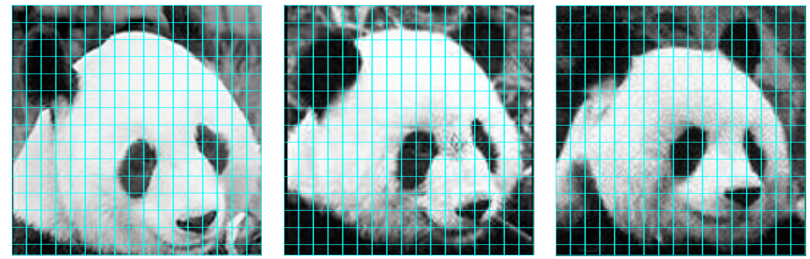




Feature Extraction: Global Appearance

- Pixel-based representations are sensitive to small shifts



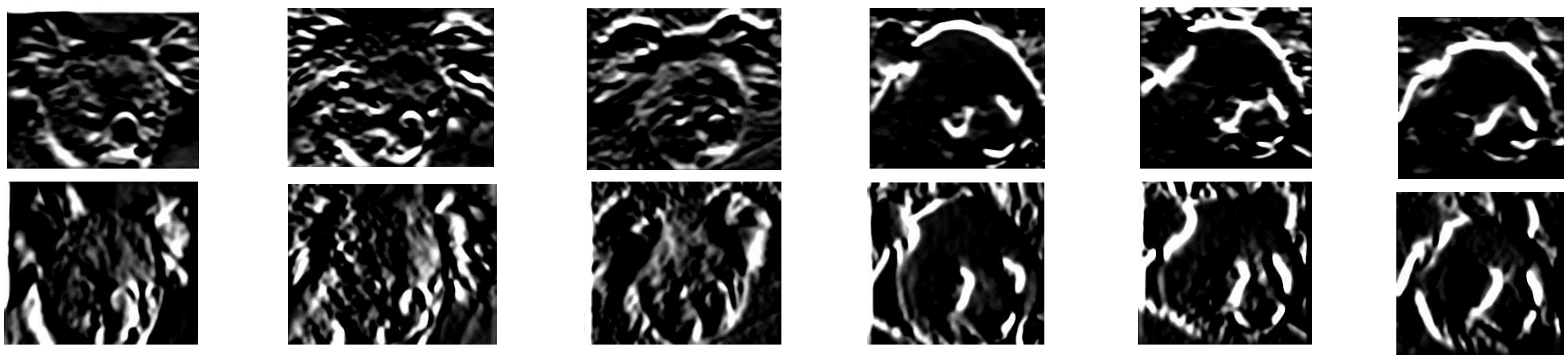


- Color or grayscale-based appearance description can be sensitive to illumination and intra-class appearance variation

Cartoon example:
an albino koala

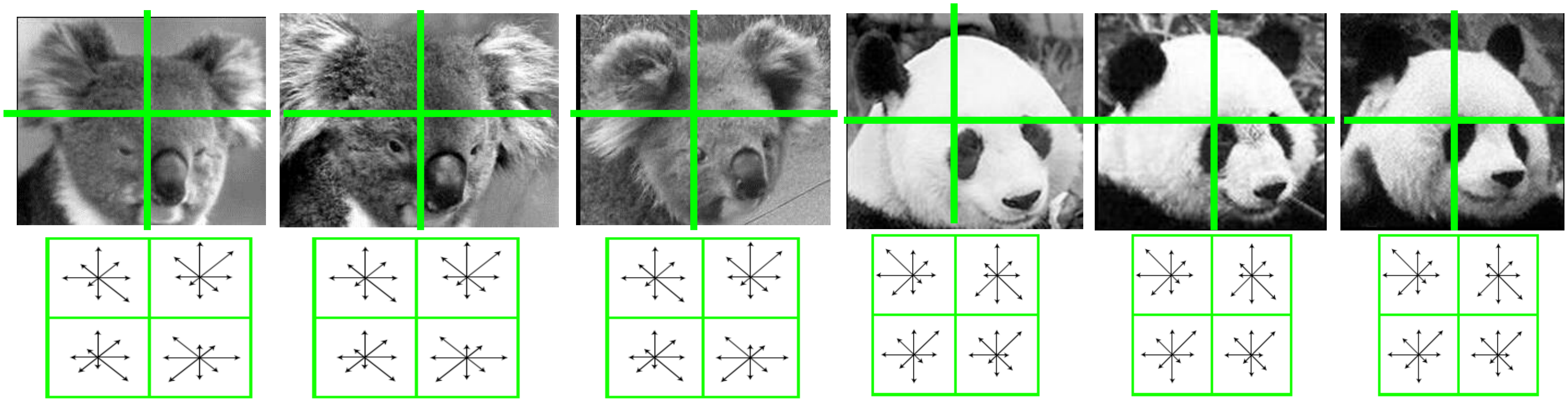
Gradient-based Representations

- Idea
  - Consider edges, contours, and (oriented) intensity gradients



Gradient-based Representations

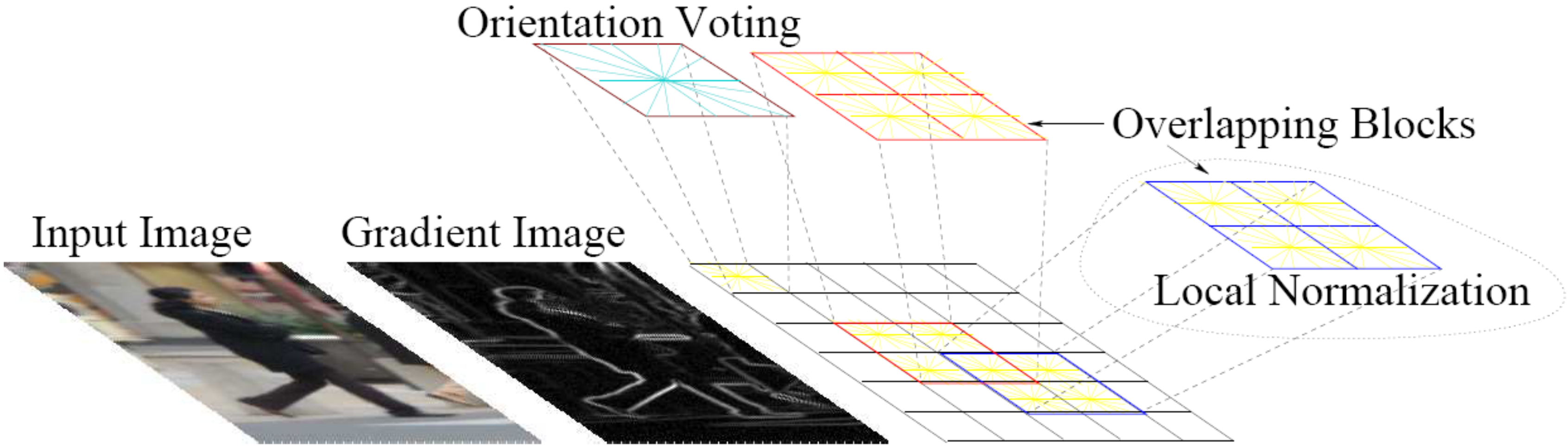
- Idea
  - Consider edges, contours, and (oriented) intensity gradients

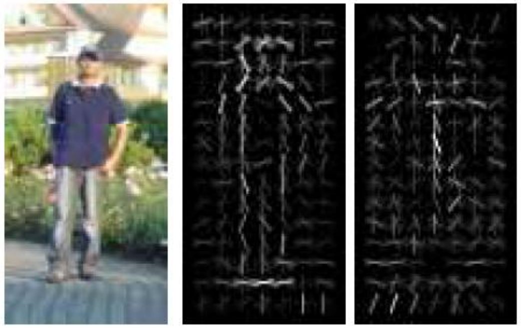


- Summarize local distribution of gradients with histograms
  - Locally orderless: offers invariance to small shifts and rotations
  - Localized histograms offer more spatial information than a single global histogram (tradeoff invariant vs. discriminative)
  - Contrast-normalization: try to correct for variable illumination

Slide credit: Kristen Grauman

Gradient-based Representations: Histograms of Oriented Gradients (HoG)





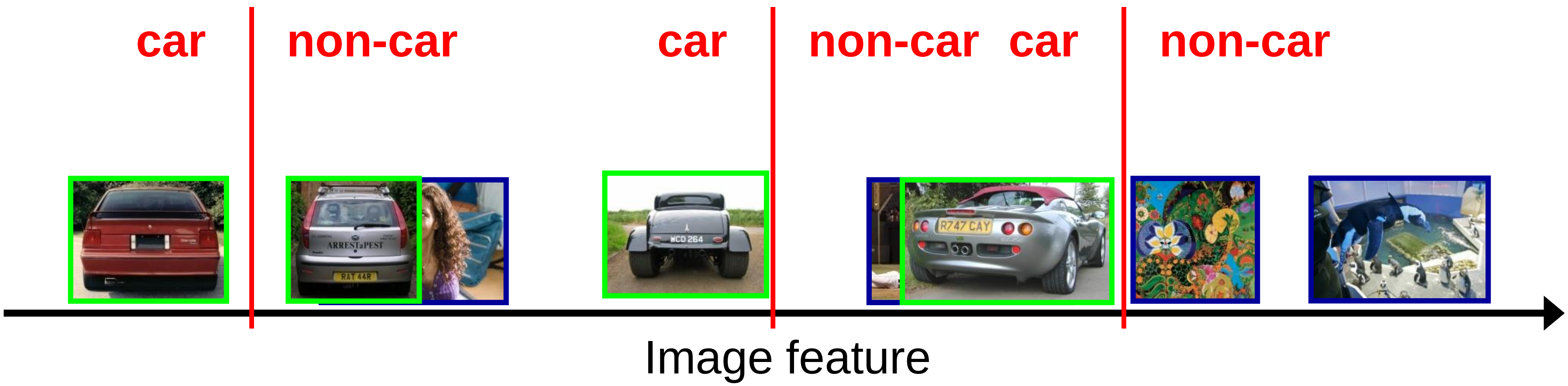
- Map each grid cell in the input window to a histogram counting the gradients per orientation.
- Code available: http://pascal.inrialpes.fr/soft/olt/

[Dalal & Triggs, CVPR 2005]

Slide credit: Kristen Grauman

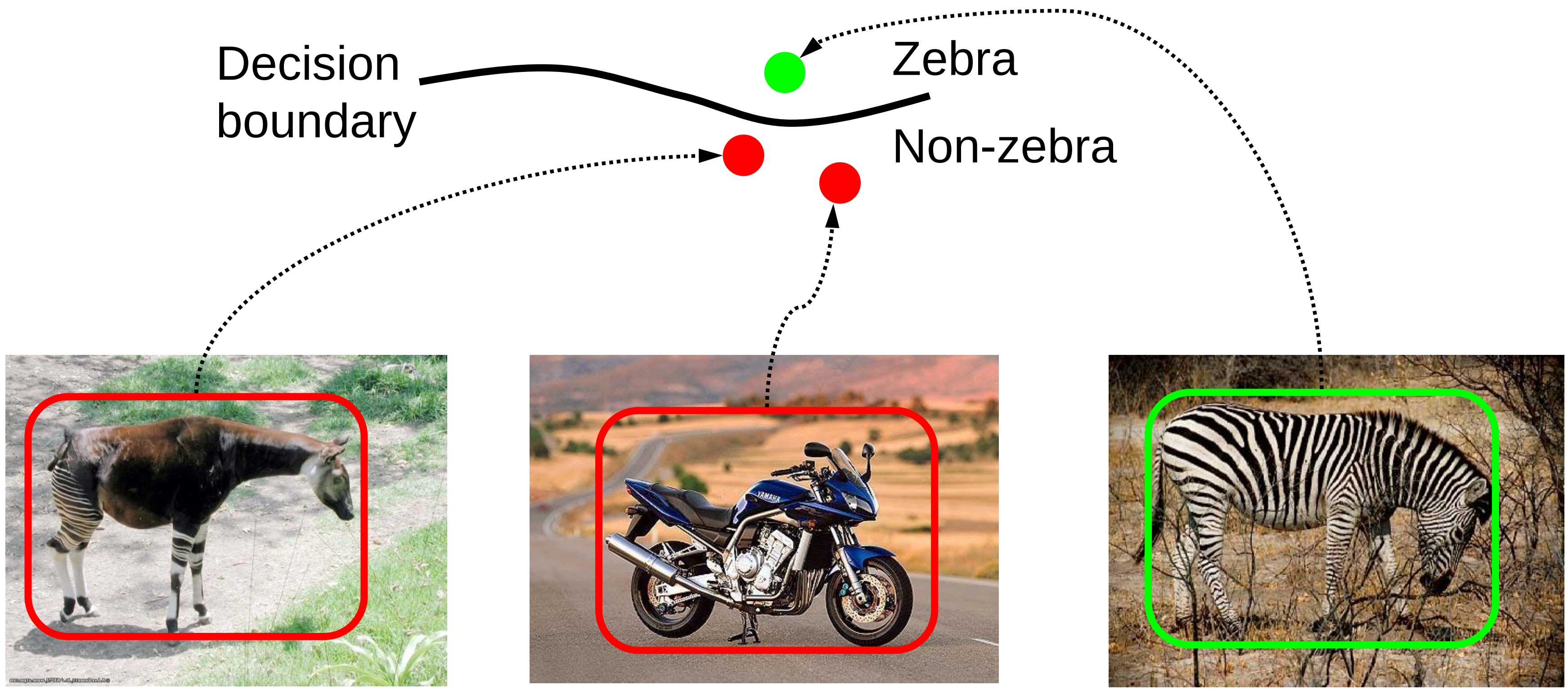
Classifier Construction

- How to compute a decision for each subwindow?



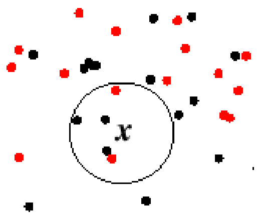
Discriminative Methods

- Learn a decision rule (classifier) assigning image features to different classes



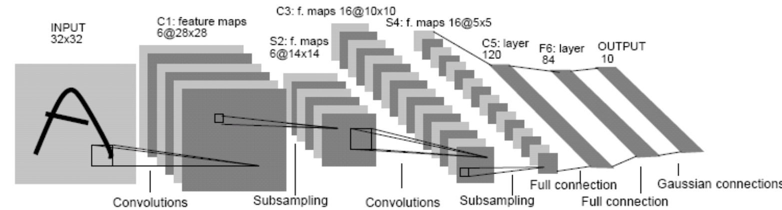
Classifier Construction: Many Choices…

Nearest Neighbor



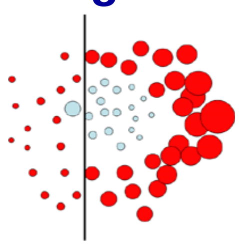
Berg, Berg, Malik 2005, Chum, Zisserman 2007, Boiman, Shechtman, Irani 2008, …

Neural networks



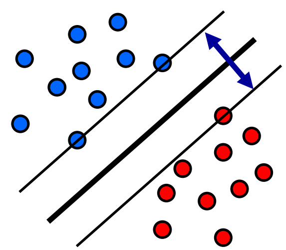
LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998
…

Boosting



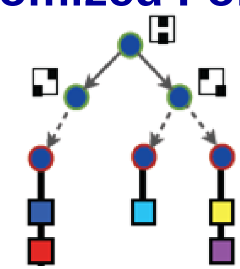
Viola, Jones 2001, Torralba et al. 2004, Opelt et al. 2006, Benenson 2012, …

Support Vector Machines



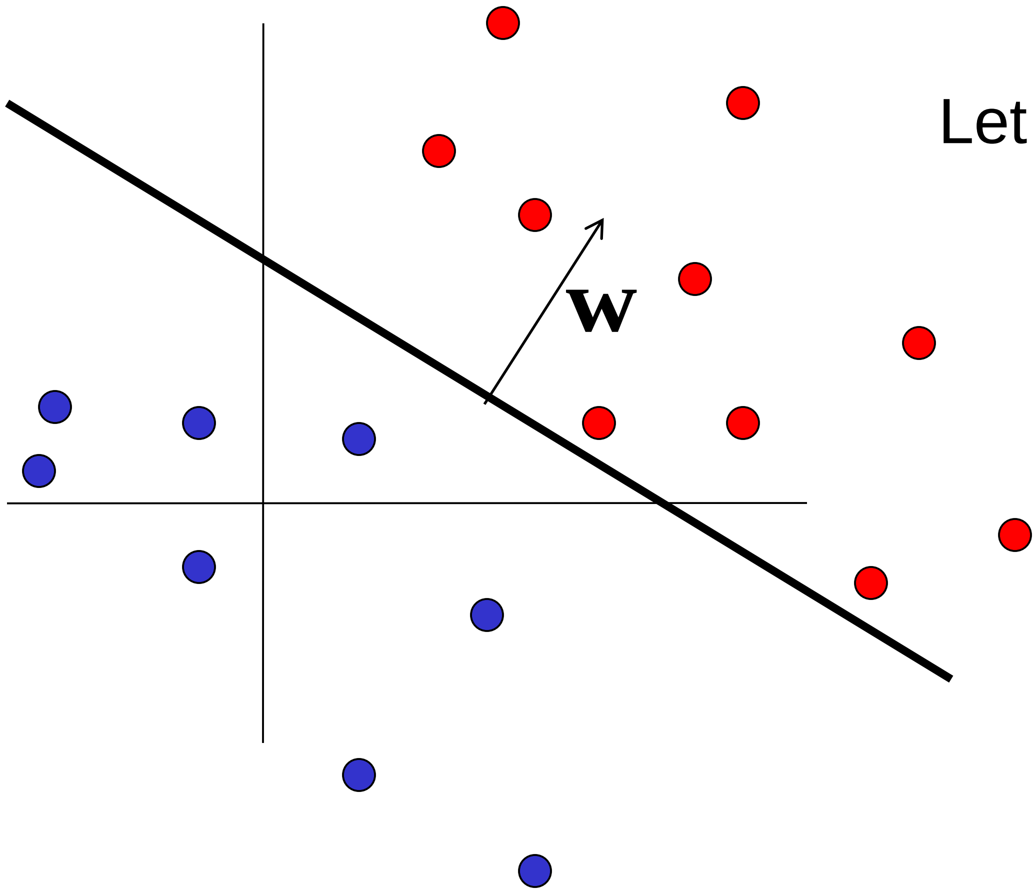
Vapnik, Schölkopf 1995, Papageorgiou, Poggio '01, Dalal, Triggs 2005, Vedaldi, Zisserman 2012

Randomized Forests



Amit, Geman 1997, Breiman 2001, Lepetit, Fua 2006, Gall, Lempitsky 2009,…

Linear Classifiers



Let $\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

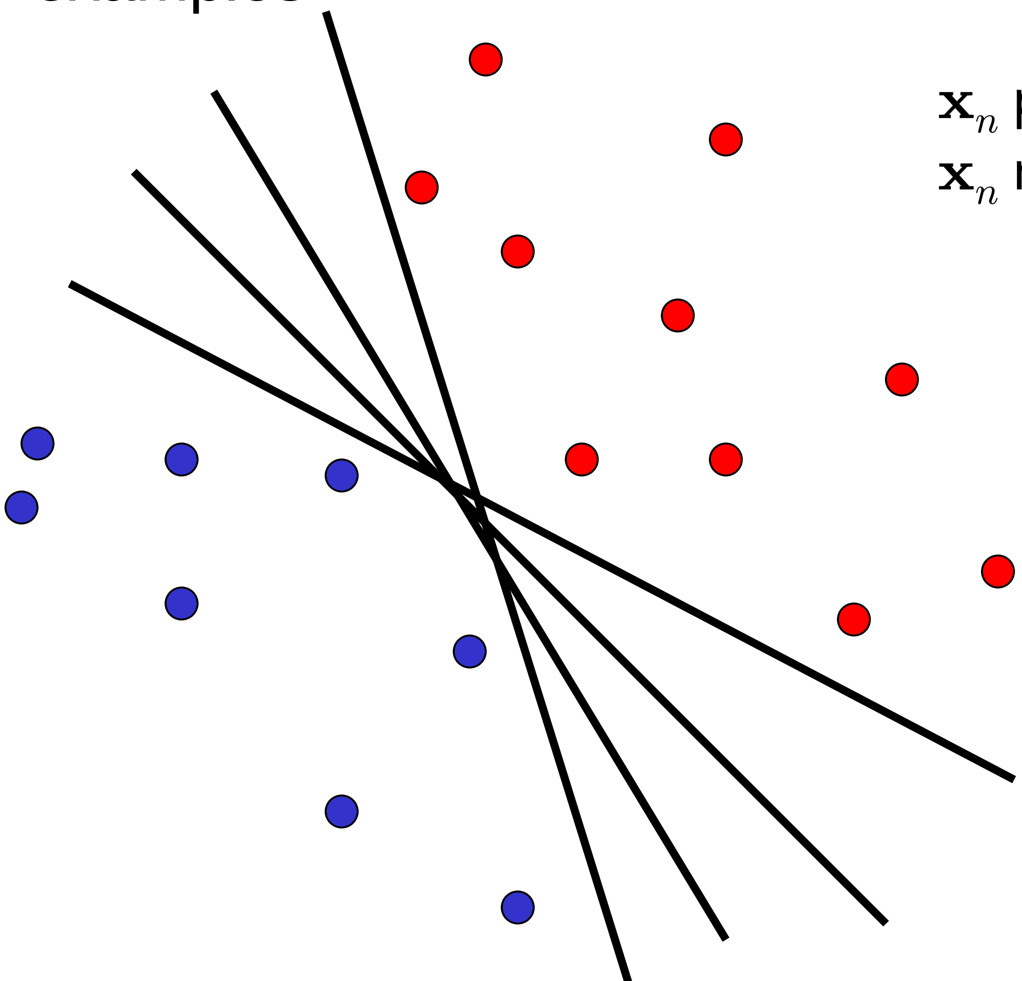
$$w_1 x_1 + w_2 x_2 + b = 0$$

$$\Updownarrow$$

$$\mathbf{w}^T \mathbf{x} + b = 0$$

Linear Classifiers

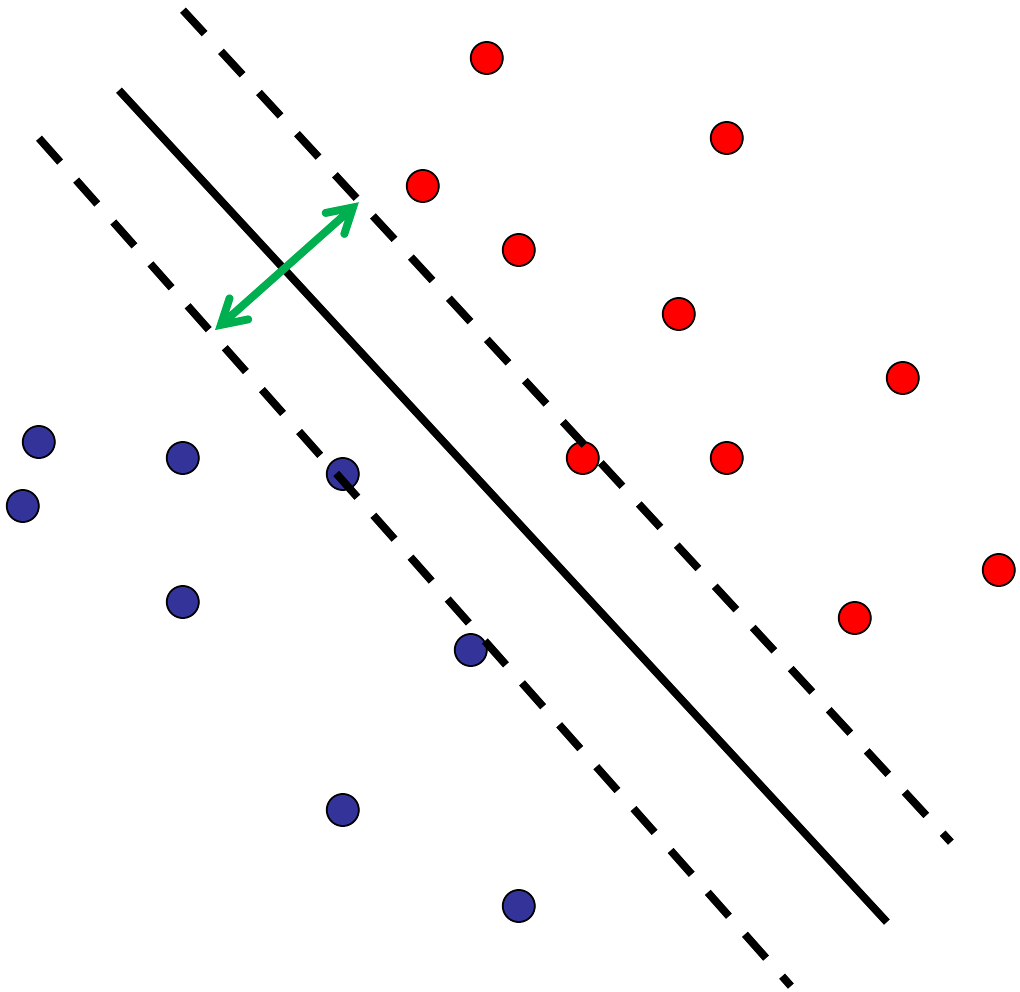
- Find linear function to separate positive and negative examples



$\mathbf{x}_n$ positive: $\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b \geq 0$
$\mathbf{x}_n$ negative: $\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b < 0$

Which line is best?

Support Vector Machines (SVMs)



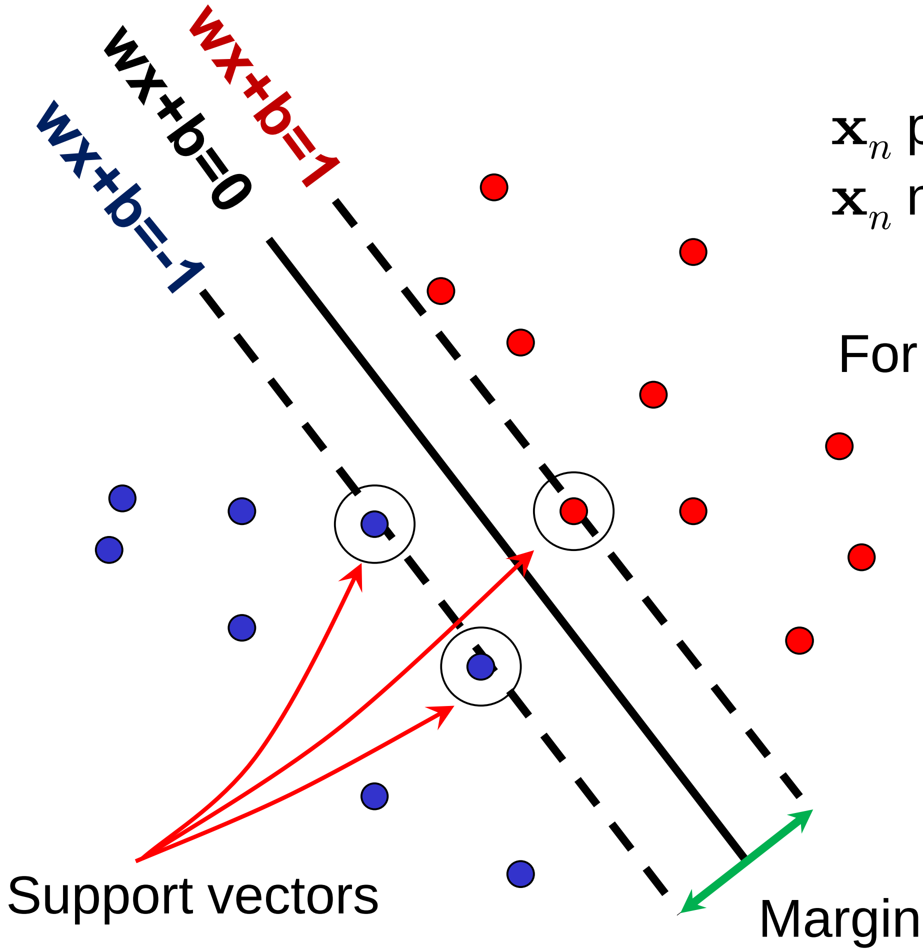
- Discriminative classifier based on *optimal separating hyperplane (i.e. line for 2D case)*
- Maximize the *margin* between the positive and negative training examples

Slide credit: Kristen Grauman

Support Vector Machines

see lecture Machine Learning!

- Want line that maximizes the margin.



$\mathbf{x}_n$ positive $(t_n = 1)$: $\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b \geq 1$

$\mathbf{x}_n$ negative $(t_n = -1)$: $\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b < -1$

For support, vectors, $\mathbf{w}^T\mathbf{x}_n + b = \pm 1$

Quadratic optimization problem

Minimize $\frac{1}{2}\mathbf{w}^T\mathbf{w}$

Subject to $t_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1$

Packages available for that…

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Finding the Maximum Margin Line

- Solution: $$\mathbf{w} = \sum_{n=1}^{N} \underbrace{a_n}_{\text{Learned weight}} t_n \underbrace{\mathbf{x}_n}_{\text{Support vector}}$$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Finding the Maximum Margin Line

- Solution: $$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \mathbf{x}_n$$

- Classification function:

$$\begin{aligned} f(\mathbf{x}) &= \mathrm{sign}(\mathbf{w}^T\mathbf{x} + b) \\ &= \mathrm{sign}\left(\sum_{n=1}^{N} a_n t_n \boxed{\mathbf{x}_n^T \mathbf{x}} + b\right) \end{aligned}$$

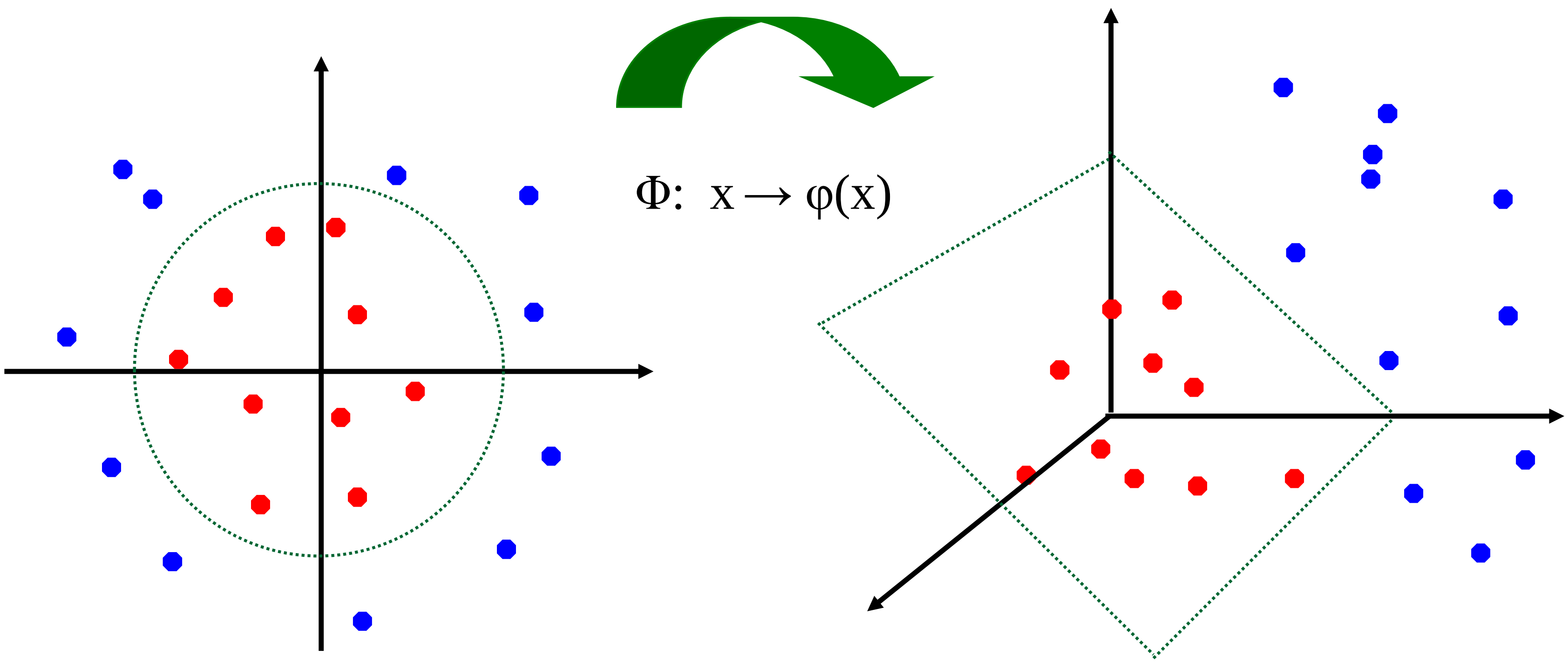
If f(x) < 0, classify as neg., if f(x) > 0, classify as pos.

  - Notice that this relies on an *inner product* between the test point $\mathbf{x}$ and the support vectors $\mathbf{x}_n$
  - (Solving the optimization problem also involves computing the inner products $\mathbf{x}_n{}^T\mathbf{x}_m$ between all pairs of training points)

C. Burges, [A Tutorial on Support Vector Machines for Pattern Recognition](), Data Mining and Knowledge Discovery, 1998

Extension: Non-Linear SVMs

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



More on that in the Machine Learning lecture…

Slide from Andrew Moore's tutorial: http://www.autonlab.org/tutorials/svm.html

Nonlinear SVMs

- *The kernel trick*: instead of explicitly computing the lifting transformation $\varphi(\mathbf{x})$, define a kernel function K such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)$$

- This gives a nonlinear decision boundary in the original feature space:

$$\sum_n a_n t_n K(\mathbf{x}_n, \mathbf{x}) + b$$

- Since the optimization formulation uses the data points only in the form of inner products $\varphi(\mathbf{x}_n)^T\varphi(\mathbf{x}_m)$, we never need to actually compute the lifting transformation $\varphi(\mathbf{x})$.

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Some Often-Used Kernel Functions

- Linear:

$$K(\mathrm{x_i},\mathrm{x_j}) = \mathrm{x_i}^{\mathrm{T}}\mathrm{x_j}$$

- Polynomial of power p:

$$K(\mathrm{x_i},\mathrm{x_j}) = (1 + \mathrm{x_i}^{\mathrm{T}}\mathrm{x_j})^p$$

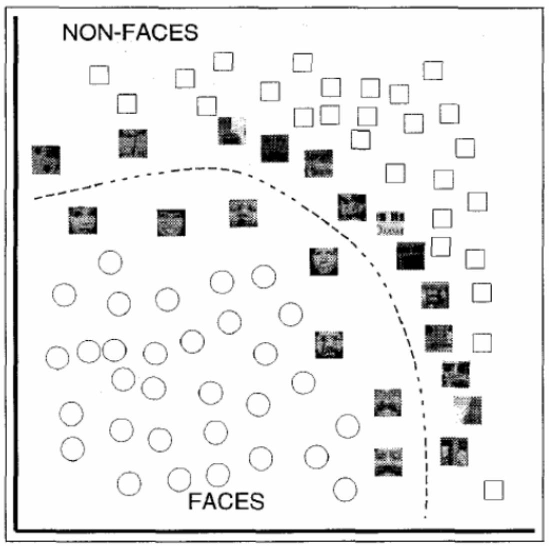
- Gaussian (Radial-Basis Function):

$$K(\mathbf{x_i},\mathbf{x_j}) = \exp\left(-\frac{\left\|\mathbf{x_i}-\mathbf{x_j}\right\|^2}{2\sigma^2}\right)$$

Slide from Andrew Moore's tutorial: http://www.autonlab.org/tutorials/svm.html

Summary: SVMs for Recognition

1. Define your representation for each example.
2. Select a kernel function.
3. Compute pairwise kernel values between labeled examples
4. Pass this "kernel matrix" to SVM optimization software to identify support vectors & weights.
5. To classify a new example: compute kernel values between new input and support vectors, apply weights, check sign of output.



Topics of This Lecture

- Object Categorization
  - Problem Definition
  - Challenges
- Sliding-Window based Object Detection
  - Detection via Classification
  - Global Representations
  - Classifier Construction
- Classification with SVMs
  - Support Vector Machines
  - HOG Detector
- Classification with Boosting
  - AdaBoost
  - Viola-Jones Face Detection

HOG Descriptor Processing Chain

Image Window

Slide adapted from Navneet Dalal

HOG Descriptor Processing Chain

- Optional: Gamma compression
  - Goal: Reduce effect of overly strong gradients
  - Replace each pixel color/intensity by its square-root

$$x \mapsto \sqrt{x}$$

$\Rightarrow$ Small performance improvement

Gamma compression

↑

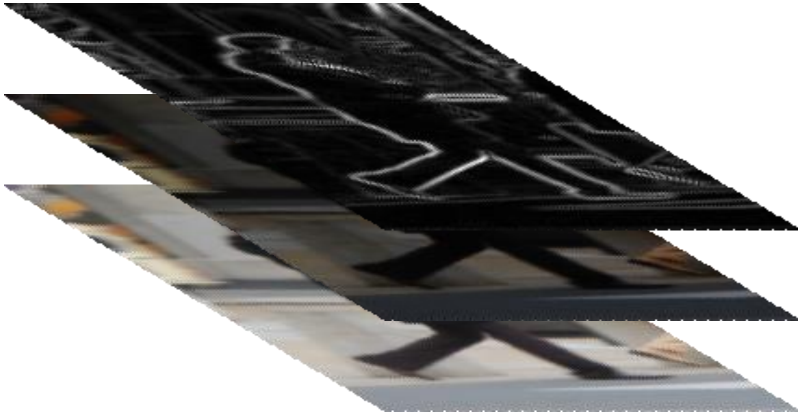
Image Window

Slide adapted from Navneet Dalal

HOG Descriptor Processing Chain

- Gradient computation
  - Compute gradients on all color channels and take strongest one
  - Simple finite difference filters work best (no Gaussian smoothing)

$$\begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$



Compute gradients

↑

Gamma compression

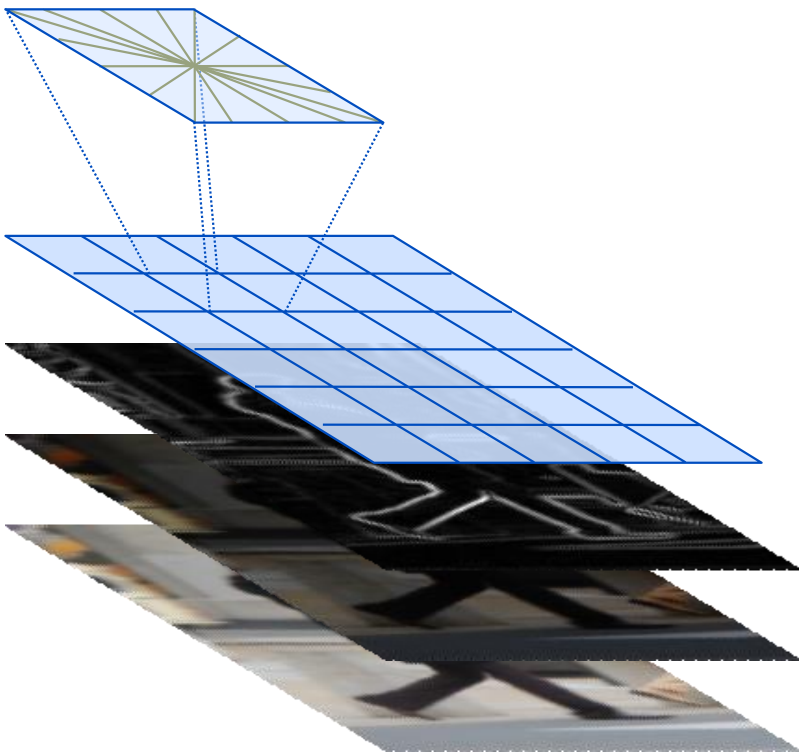
↑

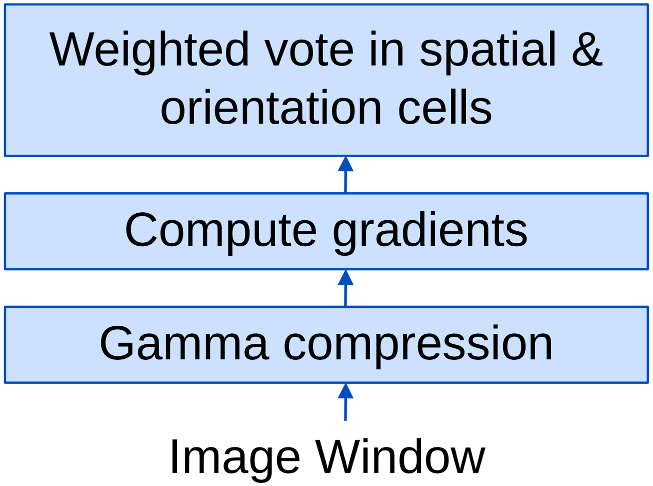
Image Window

Slide adapted from Navneet Dalal

HOG Descriptor Processing Chain

- Spatial/Orientation binning
  - Compute localized histograms of oriented gradients
  - Typical subdivision:
    8×8 cells with 8 or 9 orientation bins

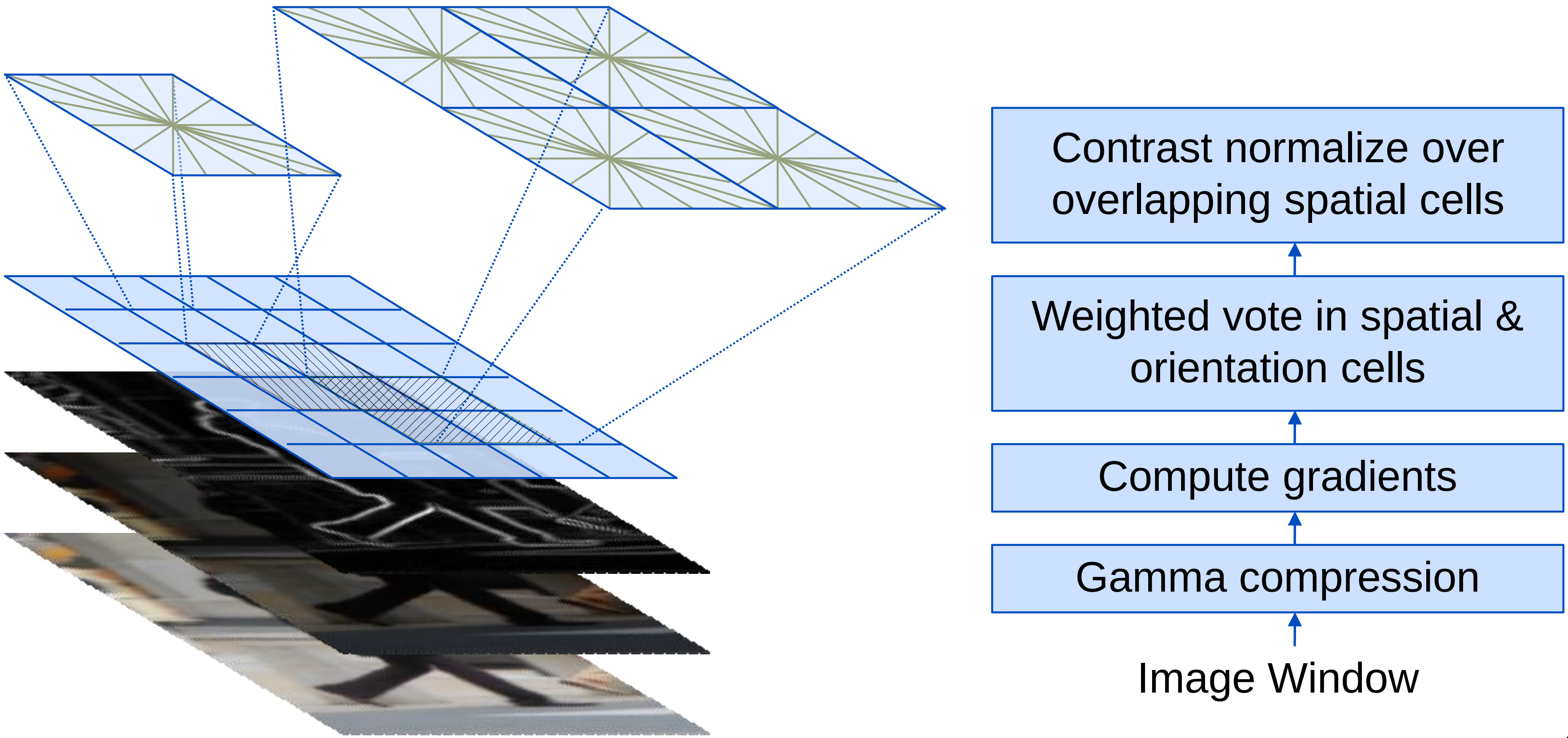




Slide adapted from Navneet Dalal

HOG Descriptor Processing Chain

- 2-Stage contrast normalization
  - L2 normalization, clipping, L2 normalization

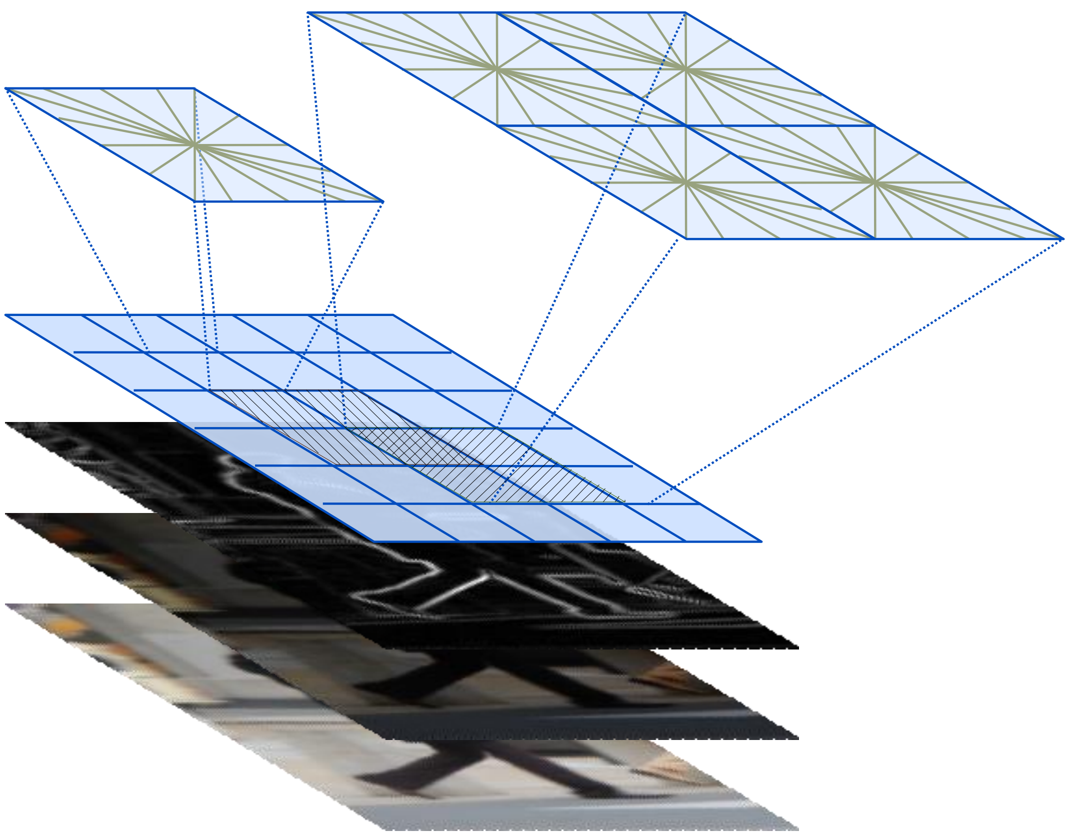


Slide adapted from Navneet Dalal

HOG Descriptor Processing Chain

- Feature vector construction
  - Collect HOG blocks into vector

[..., ..., ...,   ...]



Collect HOGs over detection window

↑

Contrast normalize over overlapping spatial cells

↑

Weighted vote in spatial & orientation cells

↑

Compute gradients

↑

Gamma compression

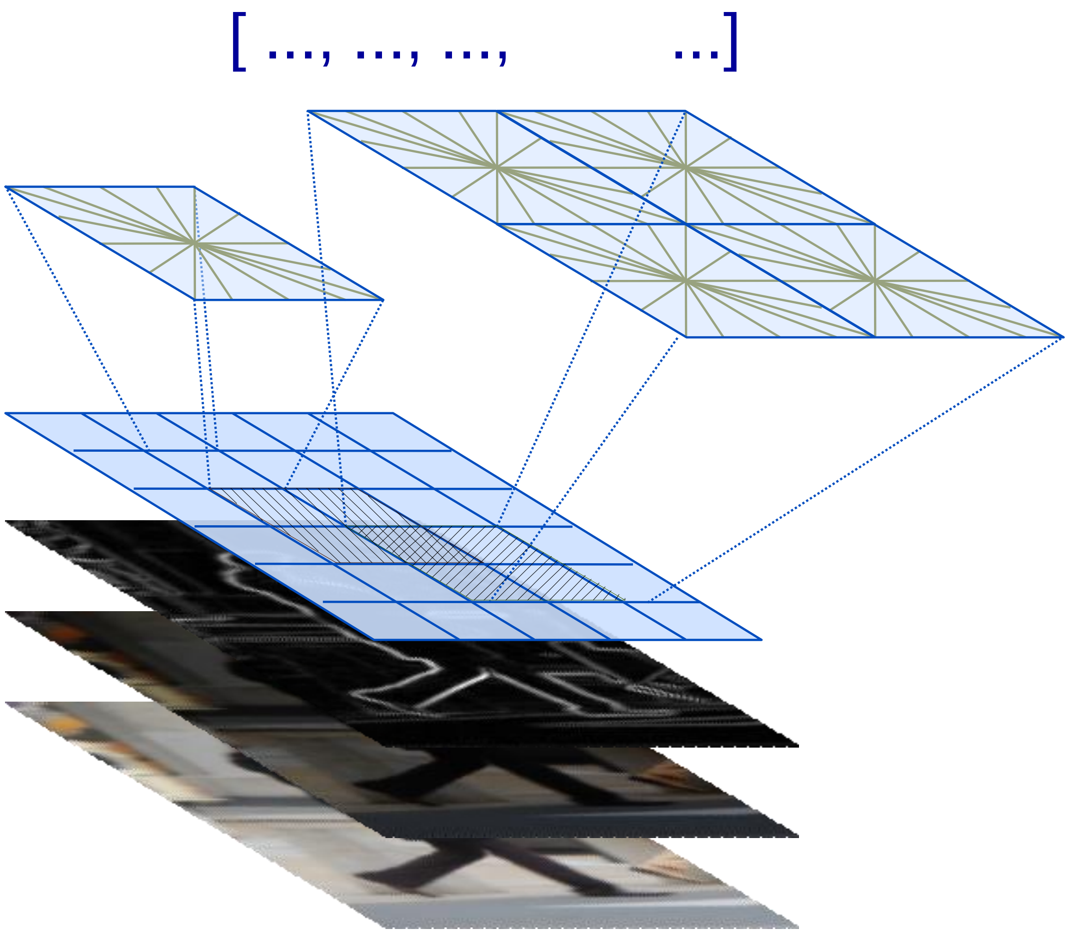
↑

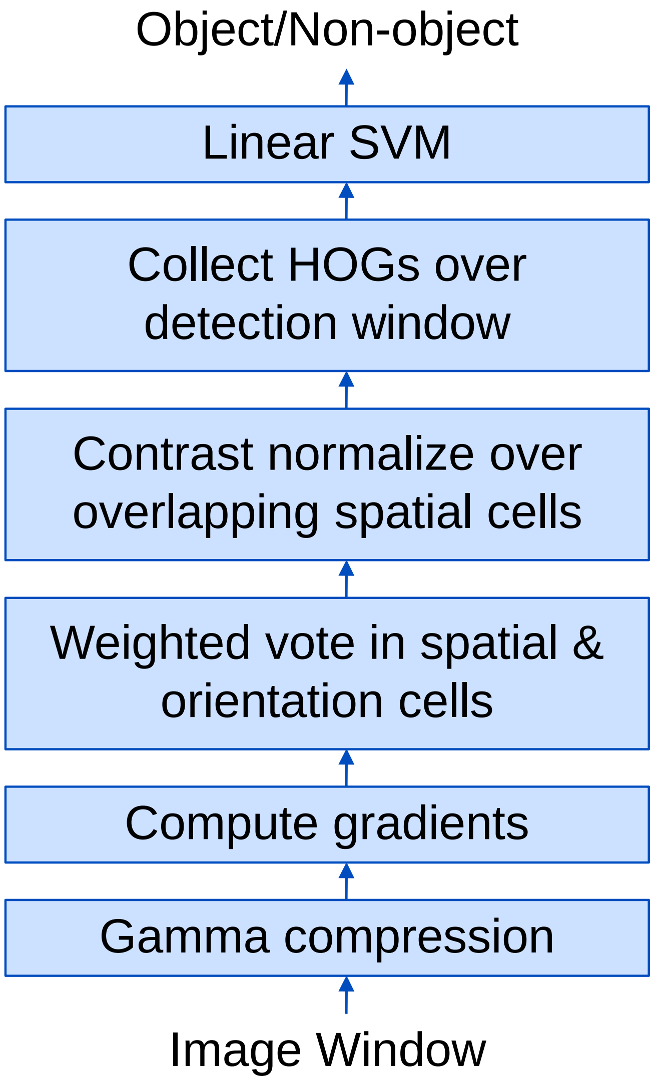
Image Window

Slide adapted from Navneet Dalal

HOG Descriptor Processing Chain

- SVM Classification
  - Typically using a linear SVM



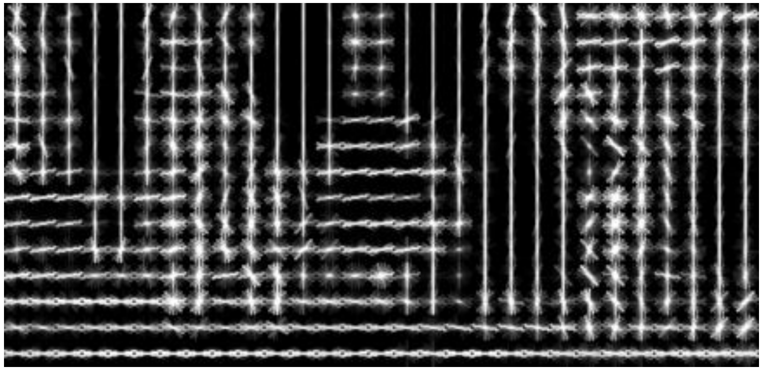


Slide adapted from Navneet Dalal

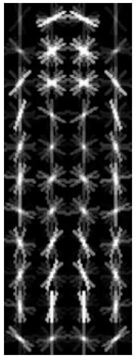
Pedestrian Detection with HOG

- Intuition
  - Train a pedestrian template using a linear SVM
  - At test time, convolve feature map with learned template $\mathbf{w}$

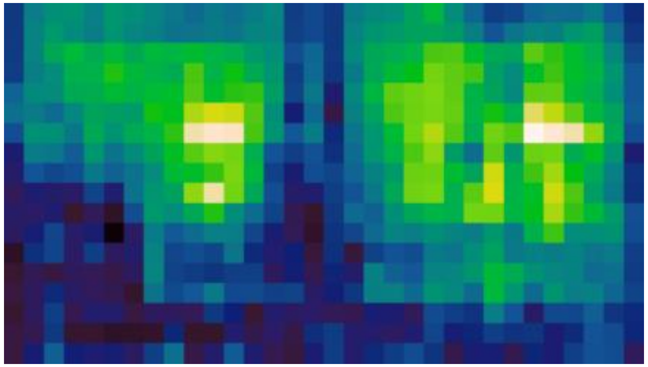
HOG feature map



Template



Detector response map



N. Dalal and B. Triggs, Histograms of Oriented Gradients for Human Detection, CVPR 2005

Slide credit: Svetlana Lazebnik

Non-Maximum Suppression

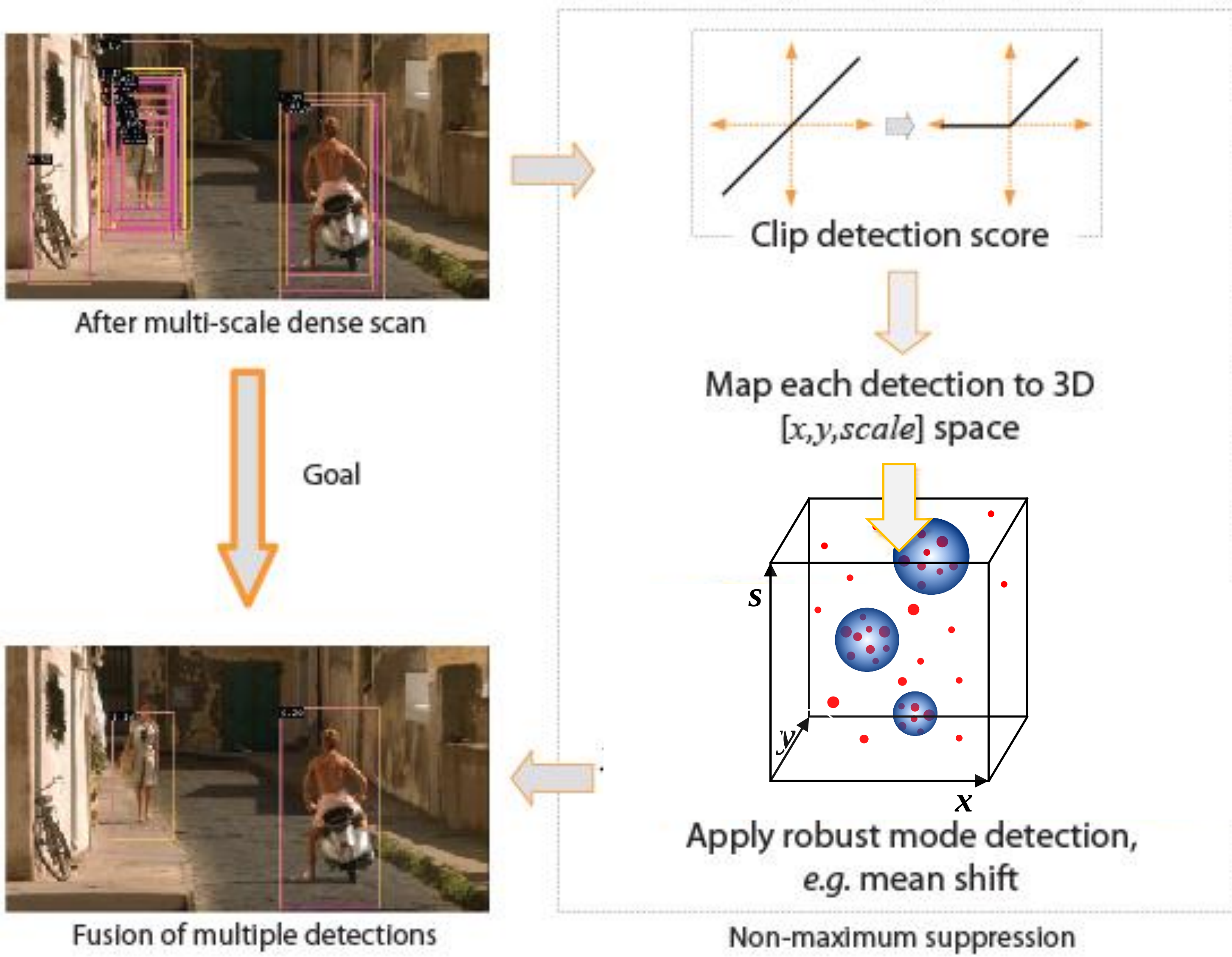


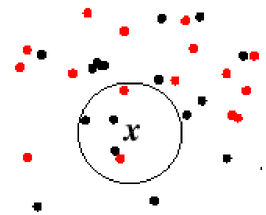
Image source: Navneet Dalal, PhD Thesis

Pedestrian detection with HoGs & SVMs

- N. Dalal, B. Triggs, Histograms of Oriented Gradients for Human Detection, CVPR 2005.

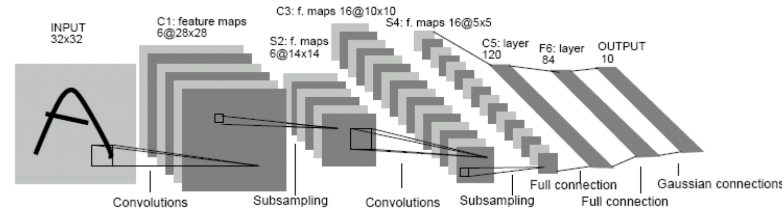
Classifier Construction: Many Choices…

Nearest Neighbor



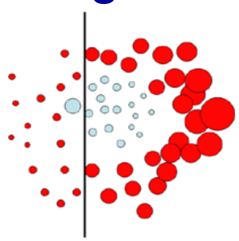
Shakhnarovich, Viola, Darrell 2003
Berg, Berg, Malik 2005,
Boiman, Shechtman, Irani 2008, …

Neural networks



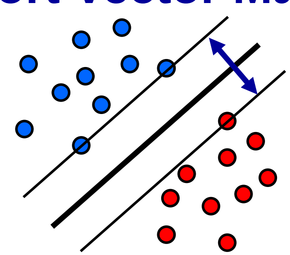
LeCun, Bottou, Bengio, Haffner 1998
Rowley, Baluja, Kanade 1998
…

Boosting



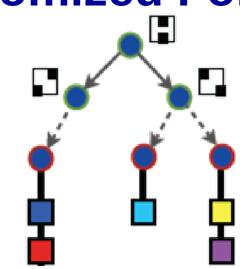
Viola, Jones 2001,
Torralba et al. 2004,
Opelt et al. 2006,
Benenson 2012, …

Support Vector Machines



Vapnik, Schölkopf 1995,
Papageorgiou, Poggio '01,
Dalal, Triggs 2005,
Vedaldi, Zisserman 2012

Randomized Forests



Amit, Geman 1997,
Breiman 2001,
Lepetit, Fua 2006,
Gall, Lempitsky 2009,…

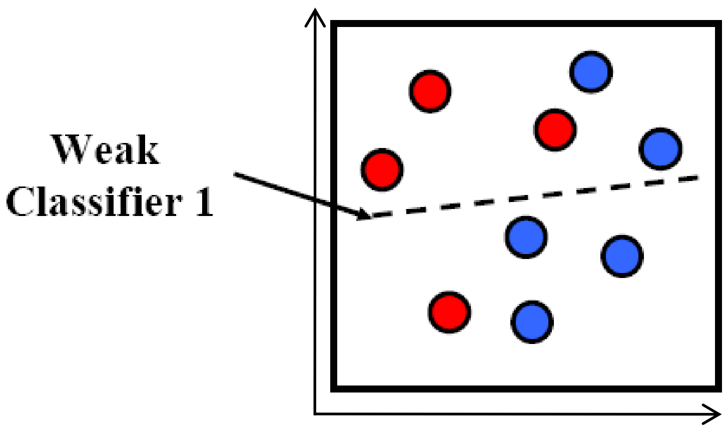
Boosting

- Idea
  - Build a strong classifier H by combining a number of "weak classifiers" $h_1,\ldots,h_M$, which need only be better than chance.
  - Sequential learning process: at each iteration, add a weak classifier
- Flexible to choice of weak learner
  - including fast simple classifiers that alone may be inaccurate
- We'll look at Freund & Schapire's AdaBoost algorithm
  - Easy to implement
  - Base learning algorithm for Viola-Jones face detector

Y. Freund and R. Schapire, [A short introduction to boosting], *Journal of Japanese Society for Artificial Intelligence*, 14(5):771-780, 1999.

Slide credit: Kristen Grauman

AdaBoost: Intuition



Consider a 2D feature space with positive and negative examples.

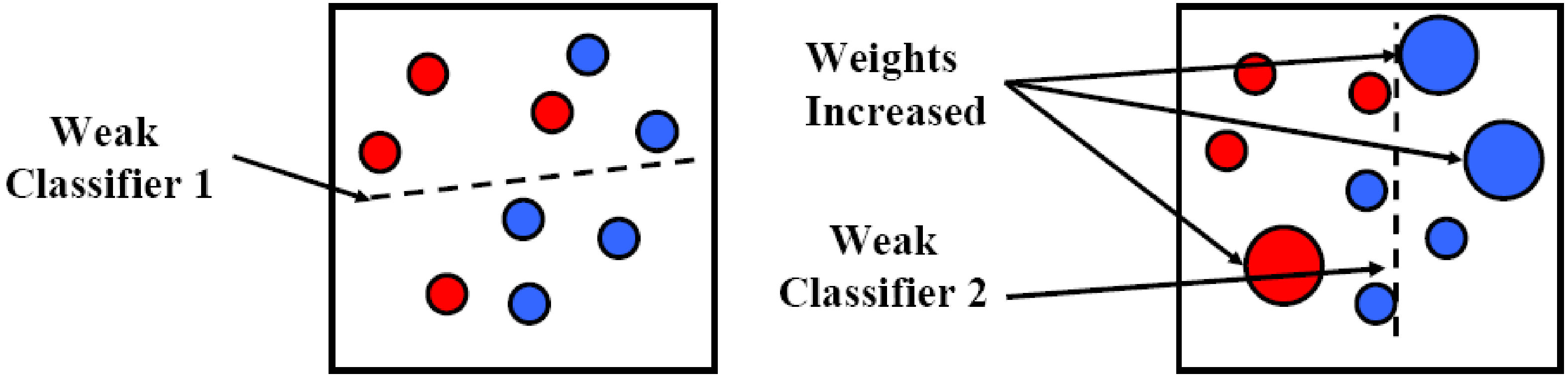
Each weak classifier splits the training examples with at least 50% accuracy.

Examples misclassified by a previous weak learner are given more emphasis at future rounds.

Figure adapted from Freund and Schapire

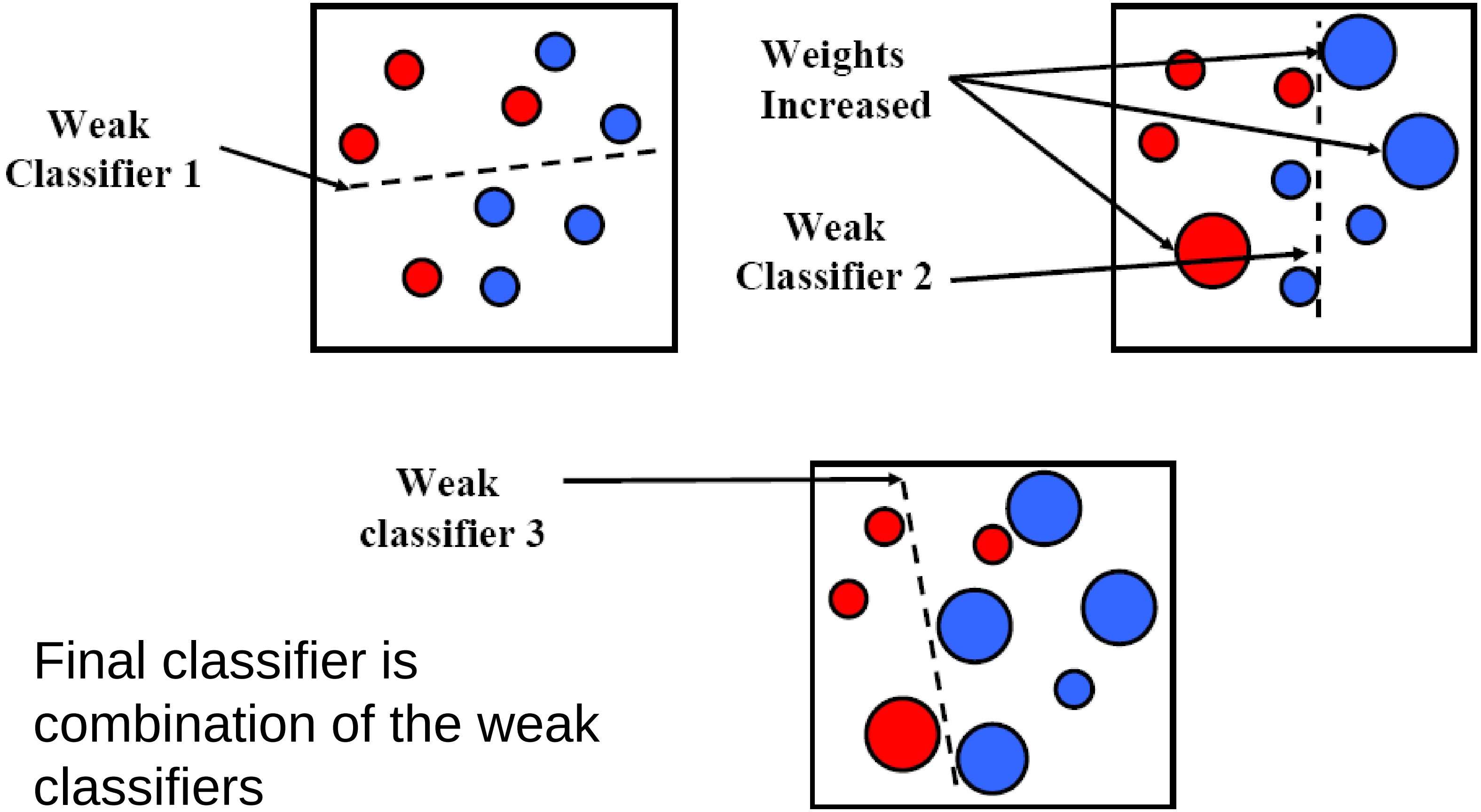
Slide credit: Kristen Grauman

AdaBoost: Intuition



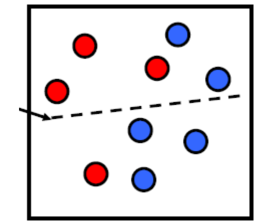
Slide credit: Kristen Grauman

AdaBoost: Intuition



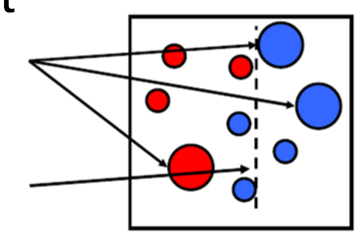
Final classifier is combination of the weak classifiers

AdaBoost – Formalization



$\{\mathbf{x}_1,\ldots,\mathbf{x}_n\}$

- 2-class classification problem
  - Given: training set $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ with target values $\mathbf{T} = \{t_1, \ldots, t_N\}$, $t_n \in \{-1,1\}$.
  - Associated weights $\mathbf{W}=\{w_1, \ldots, w_N\}$ for each training point.
- Basic steps
  - In each iteration, AdaBoost trains a new weak classifier $h_m(\mathbf{x})$ based on the current weighting coefficients $\mathbf{W}^{(m)}$.
  - We then adapt the weighting coefficients for each point
    - Increase w_n if $\mathbf{x}_n$ was misclassified by $h_m(\mathbf{x})$.
    - Decrease w_n if $\mathbf{x}_n$ was classified correctly by $h_m(\mathbf{x})$.
  - Make predictions using the final combined model



$$H(\mathbf{x}) = sign\left(\sum_{m=1}^{M} \alpha_m h_m(\mathbf{x})\right)$$

AdaBoost: Detailed Training Algorith

see lecture Machine Learning!

$\{\mathbf{x}_1,\ldots,\mathbf{x}_n\}$

1. Initialization: Set $w_n^{(1)} = \frac{1}{N}$ for $n = 1,\ldots,N$.
2. For $m = 1,\ldots,M$ iterations

   a) Train a new weak classifier $h_m(\mathbf{x})$ using the current weighting coefficients $\mathbf{W}^{(m)}$ by minimizing the weighted error function

   $$J_m = \sum_{n=1}^{N} w_n^{(m)} I(h_m(\mathbf{x}_n) \neq t_n) \qquad I(A) = \begin{cases} 1, & \text{if } A \text{ is true} \\ 0, & \text{else} \end{cases}$$

   b) Estimate the weighted error of this classifier on $\mathbf{X}$:

   $$\epsilon_m = \frac{\sum_{n=1}^{N} w_n^{(m)} I(h_m(\mathbf{x}_n) \neq t_n)}{\sum_{n=1}^{N} w_n^{(m)}}$$

   c) Calculate a weighting coefficient for $h_m(\mathbf{x})$:

   $$\alpha_m = \ln\left\{\frac{1-\epsilon_m}{\epsilon_m}\right\}$$

   d) Update the weighting coefficients:

   $$w_n^{(m+1)} = w_n^{(m)} \exp\left\{\alpha_m I(h_m(\mathbf{x}_n) \neq t_n)\right\}$$

AdaBoost: Recognition

- Evaluate all selected weak classifiers on test data.

$$h_1(\mathbf{x}), \ldots, h_m(\mathbf{x})$$

- Final classifier is weighted combination of selected weak classifiers:

$$H(\mathbf{x}) = sign\left(\sum_{m=1}^{M} \alpha_m h_m(\mathbf{x})\right)$$

- Very simple procedure!
  - Less than 10 lines in Matlab!
  - But works extremely well in practice…

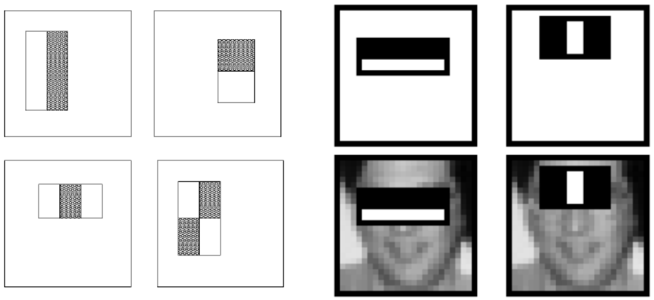
Example: Face Detection

- Frontal faces are a good example of a class where global appearance models + a sliding window detection approach fit well:
  - Regular 2D structure
  - Center of face almost shaped like a “patch”/window

- Now we’ll take AdaBoost and see how the Viola-Jones face detector works

Feature extraction

"Rectangular" filters

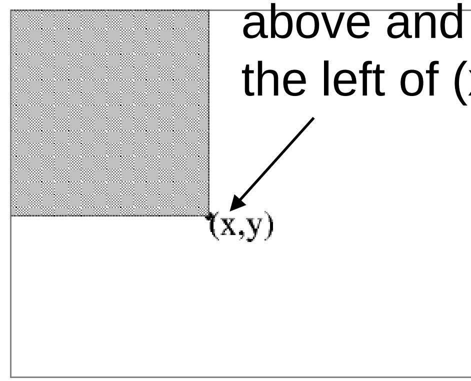


Feature output is difference between adjacent regions

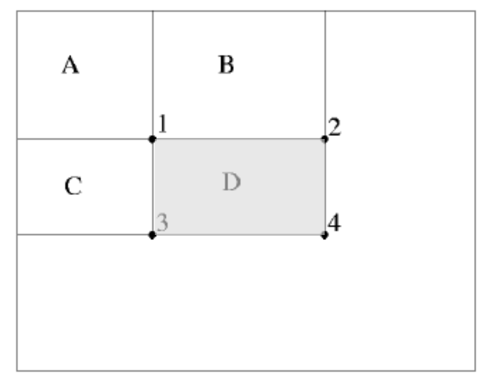
Efficiently computable with integral image: any sum can be computed in constant time

Avoid scaling images → scale features directly for same cost

Value at (x,y) is sum of pixels above and to the left of (x,y)



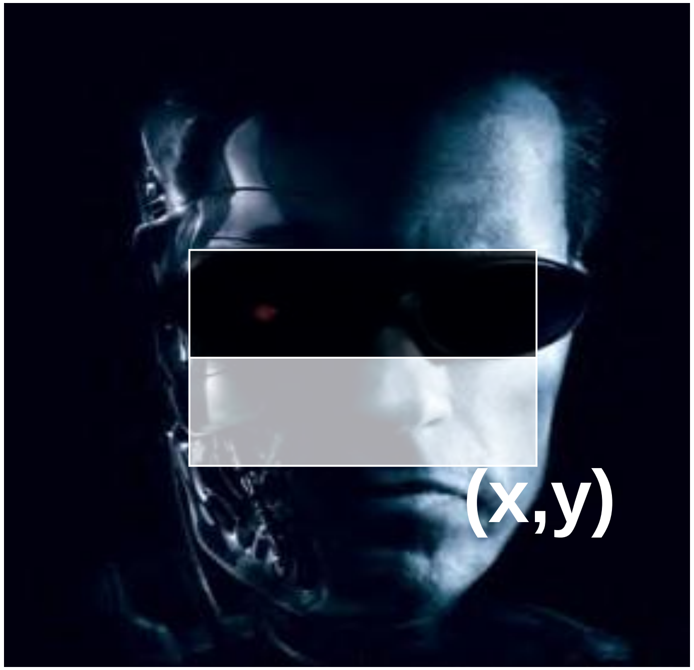
Integral image

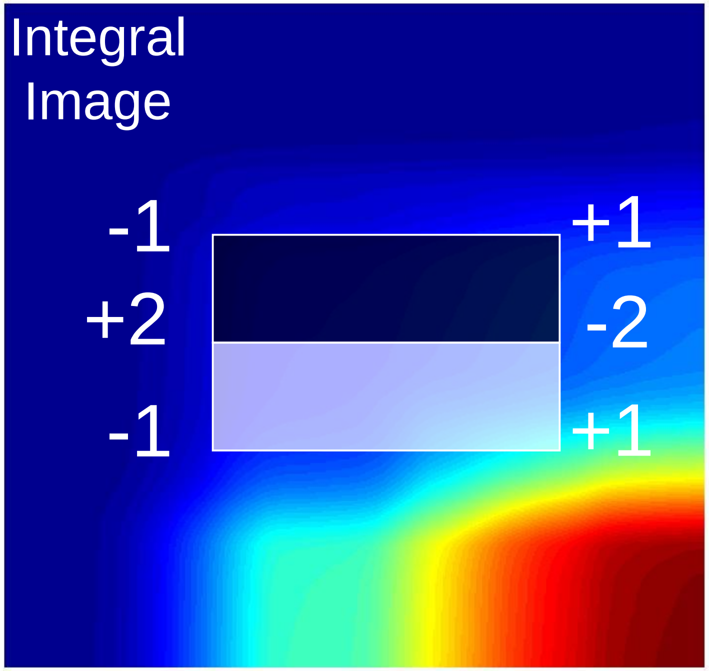


$$D = 1 + 4 - (2 + 3)$$
$$= A + (A + B + C + D) - (A + C + A + B)$$
$$= D$$

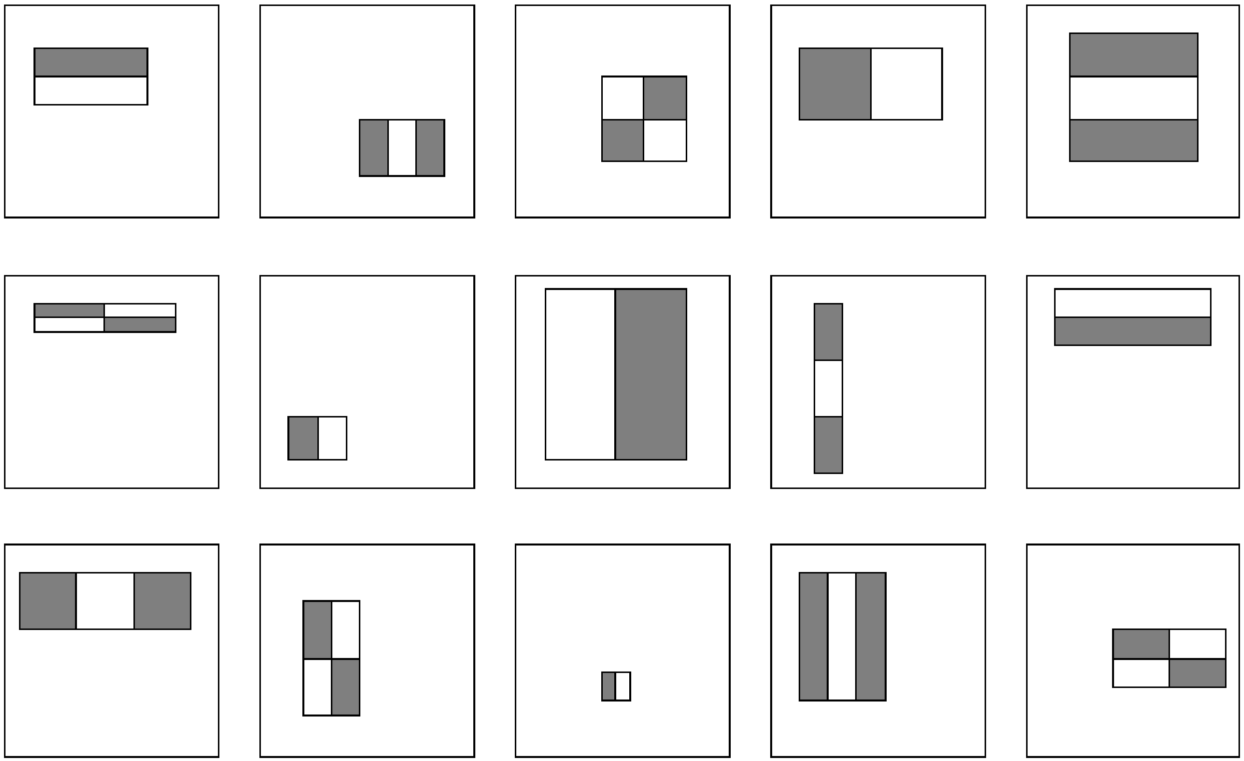
[Viola & Jones, CVPR 2001]

Example





Large Library of Features



Considering all possible filter parameters: position, scale, and type:

180,000+ possible features associated with each 24 x 24 window

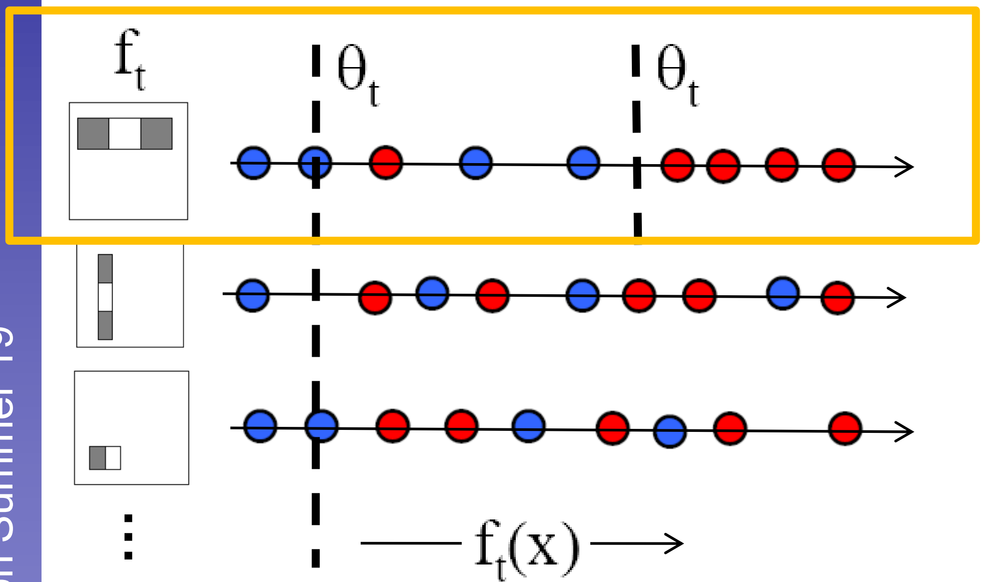
Use AdaBoost both to select the informative features and to form the classifier

Weak classifier: feature output > θ?

[Viola & Jones, CVPR 2001]

AdaBoost for Feature+Classifier Selection

- Want to select the single rectangle feature and threshold that best separates positive (faces) and negative (non-faces) training examples, in terms of *weighted* error.



Outputs of a possible rectangle feature on faces and non-faces.

Resulting weak classifier:

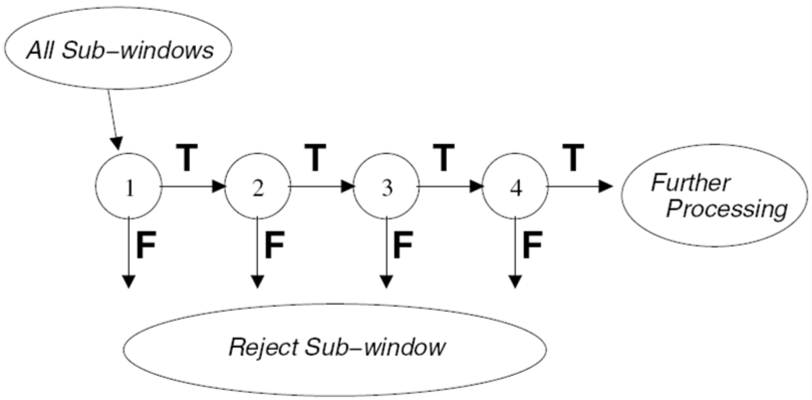
$$h_t(x) = \begin{cases} +1 & \text{if } f_t(x) > \theta_t \\ -1 & \text{otherwise} \end{cases}$$

For next round, reweight the examples according to errors, choose another filter/threshold combo.

[Viola & Jones, CVPR 2001]

Cascading Classifiers for Detection

- Even if the filters are fast to compute, each new image has a lot of possible windows to search.
- For efficiency, apply less accurate but faster classifiers first to immediately discard windows that clearly appear to be negative; e.g.,

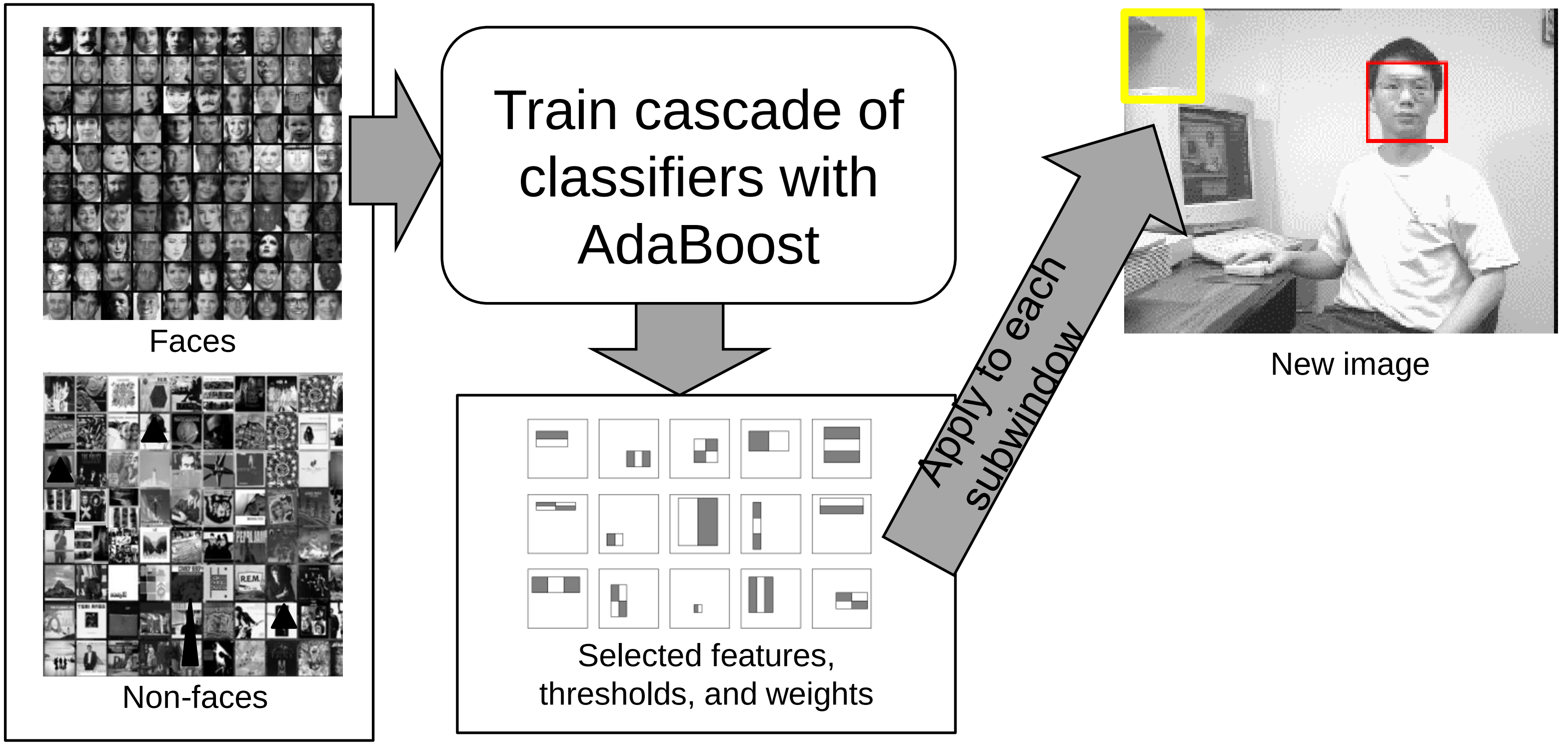


  - Filter for promising regions with an initial inexpensive classifier
  - Build a chain of classifiers, choosing cheap ones with low false negative rates early in the chain

[Fleuret & Geman, IJCV 2001]
[Rowley et al., PAMI 1998]
[Viola & Jones, CVPR 2001]

Figure from Viola & Jones CVPR

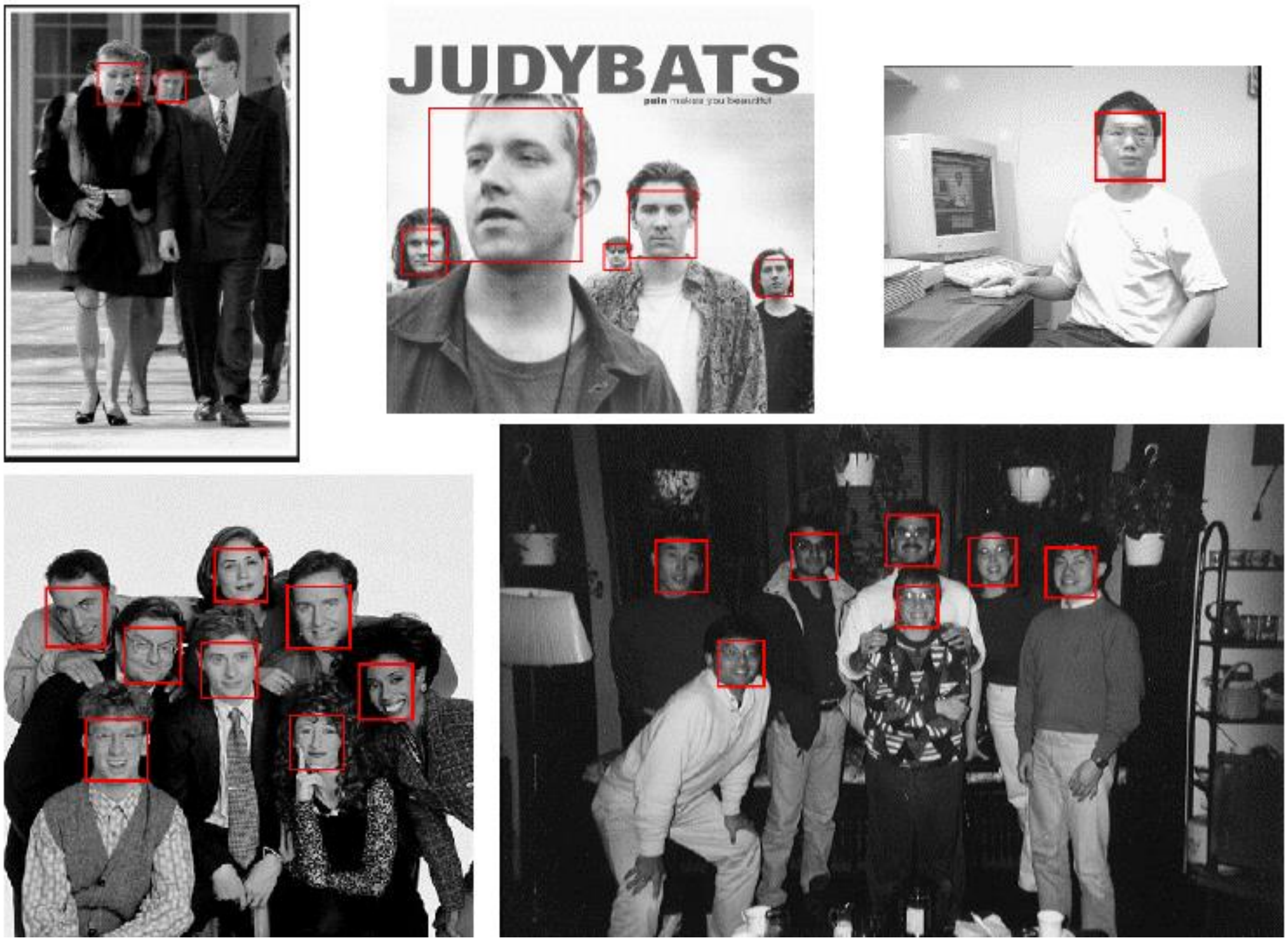
Viola-Jones Face Detector: Summary



- Train with 5K positives, 350M negatives
- Real-time detector using 38 layer cascade
- 6061 features in final layer
- [Implementation available in OpenCV: http://sourceforge.net/projects/opencvlibrary/]

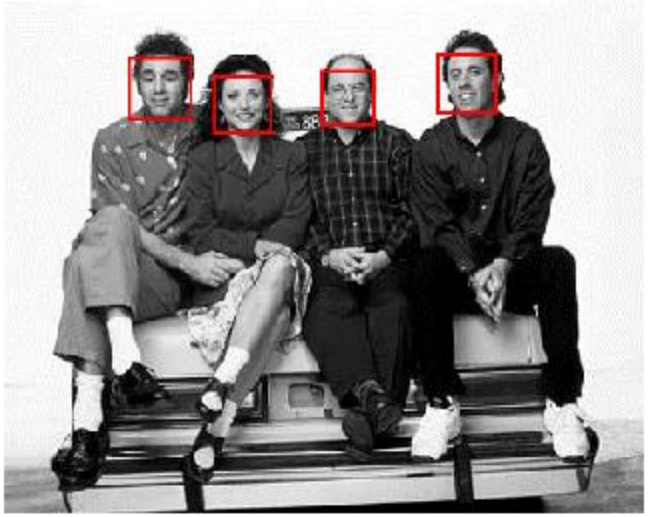
Slide credit: Kristen Grauman

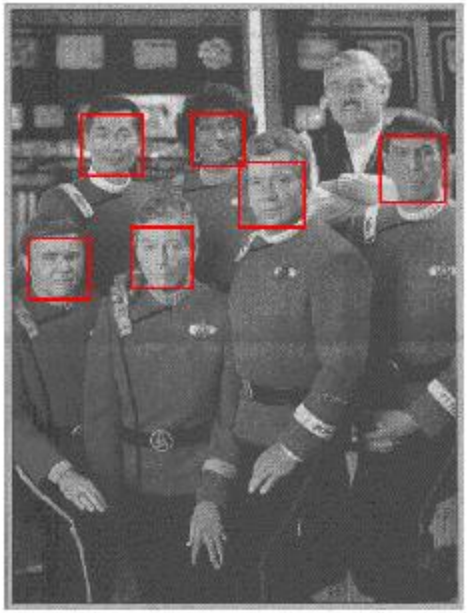
Viola-Jones Face Detector: Results

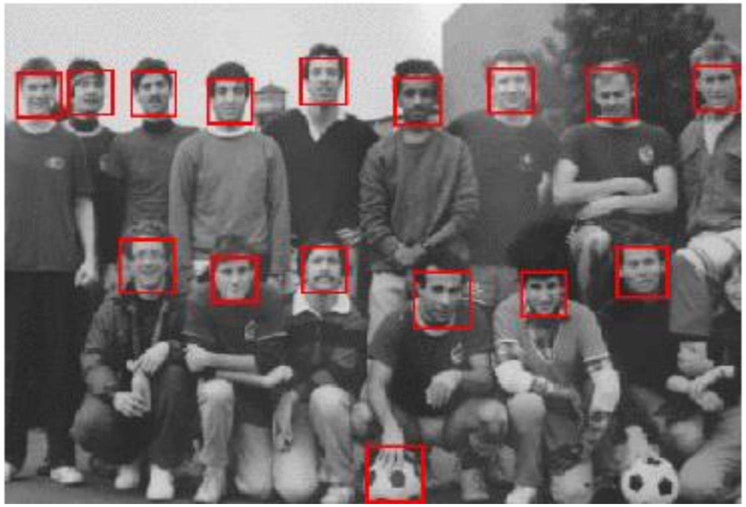


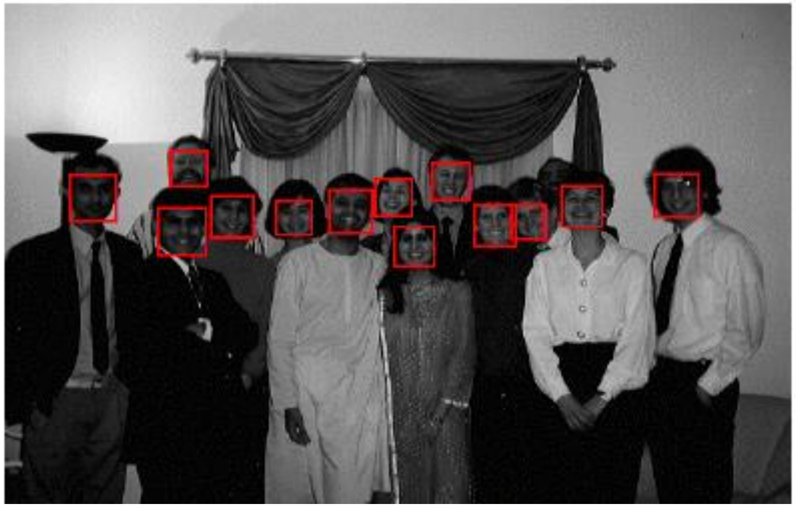
Slide credit: Kristen Grauman

Viola-Jones Face Detector: Results







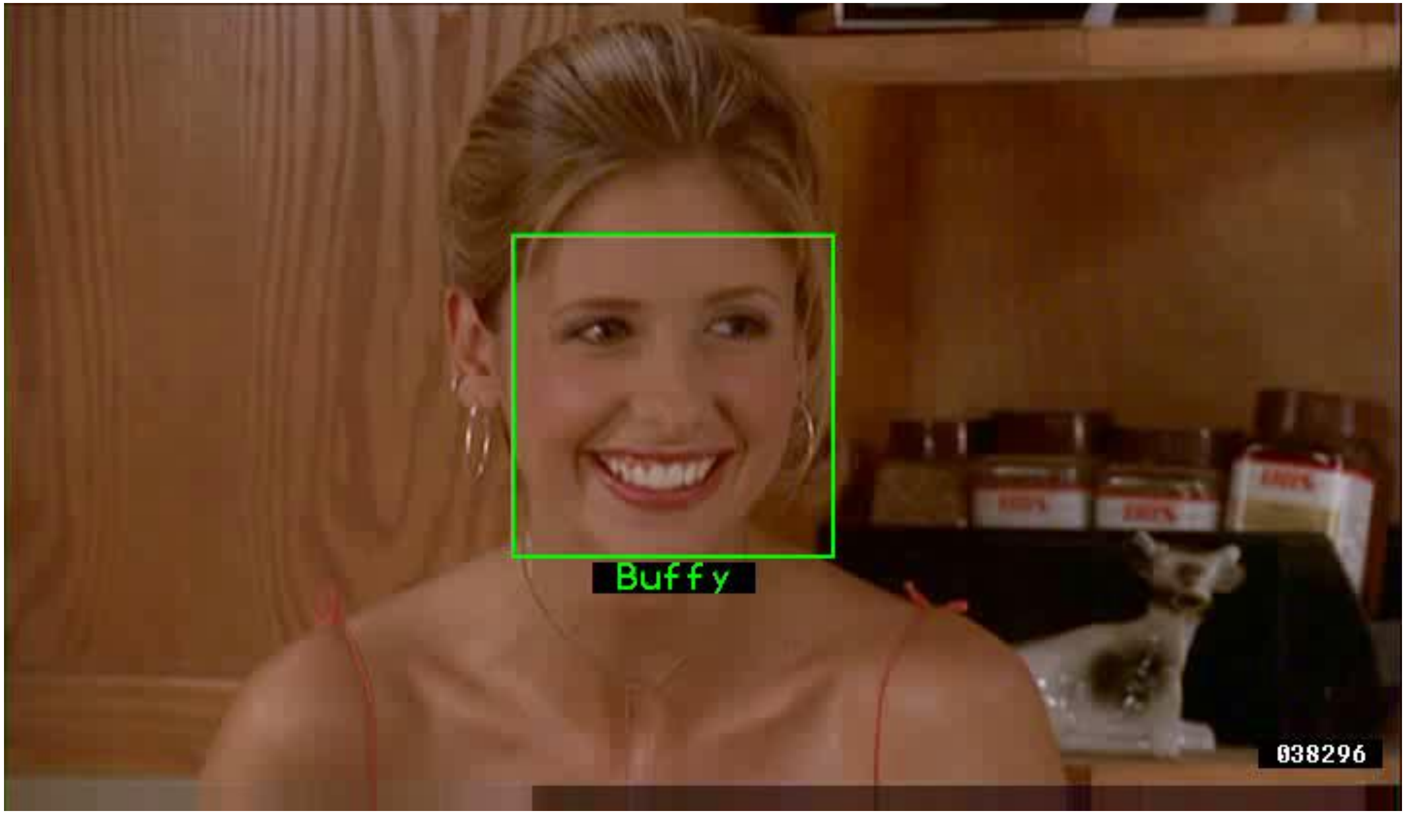


You Can Try It At Home…

- The Viola & Jones detector was a huge success
  - First real-time face detector available
  - Many derivative works and improvements
- C++ implementation available in OpenCV [Lienhart, 2002]
  - http://sourceforge.net/projects/opencvlibrary/
- Matlab wrappers for OpenCV code available, e.g. here
  - http://www.mathworks.com/matlabcentral/fileexchange/19912

P. Viola, M. Jones, Robust Real-Time Face Detection, IJCV, Vol. 57(2), 2004

Example Application



Frontal faces detected and then tracked, character names inferred with alignment of script and subtitles.

Everingham, M., Sivic, J. and Zisserman, A.
"Hello! My name is... Buffy" - Automatic naming of characters in TV video, BMVC 2006.
http://www.robots.ox.ac.uk/~vgg/research/nface/index.html

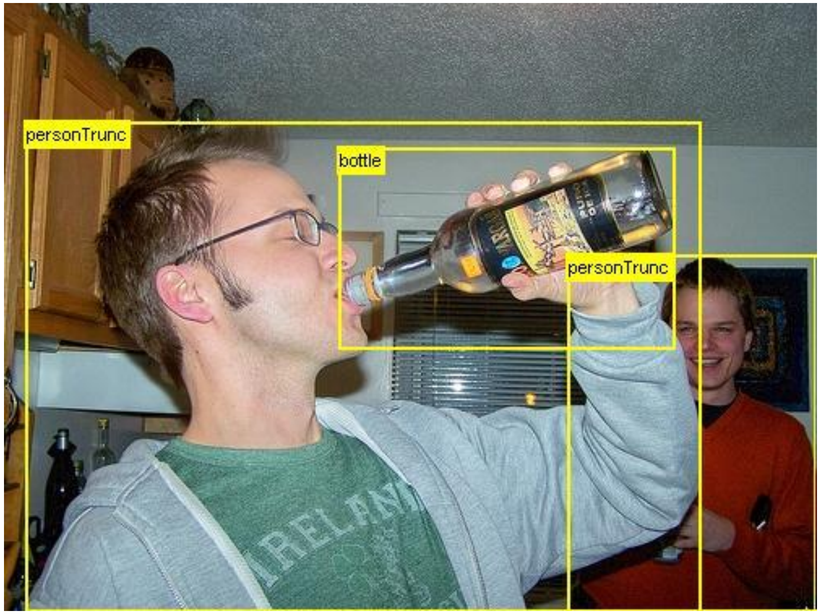
Summary: Sliding-Windows

- Pros
  - Simple detection protocol to implement
  - Good feature choices critical
  - Past successes for certain classes
  - Good detectors available (Viola & Jones, HOG, etc.)

- Cons/Limitations
  - High computational complexity
    - For example: 250,000 locations x 30 orientations x 4 scales = 30,000,000 evaluations!
    - This puts tight constraints on the classifiers we can use.
    - If training binary detectors independently, this means cost increases linearly with number of classes.
  - With so many windows, false positive rate better be low

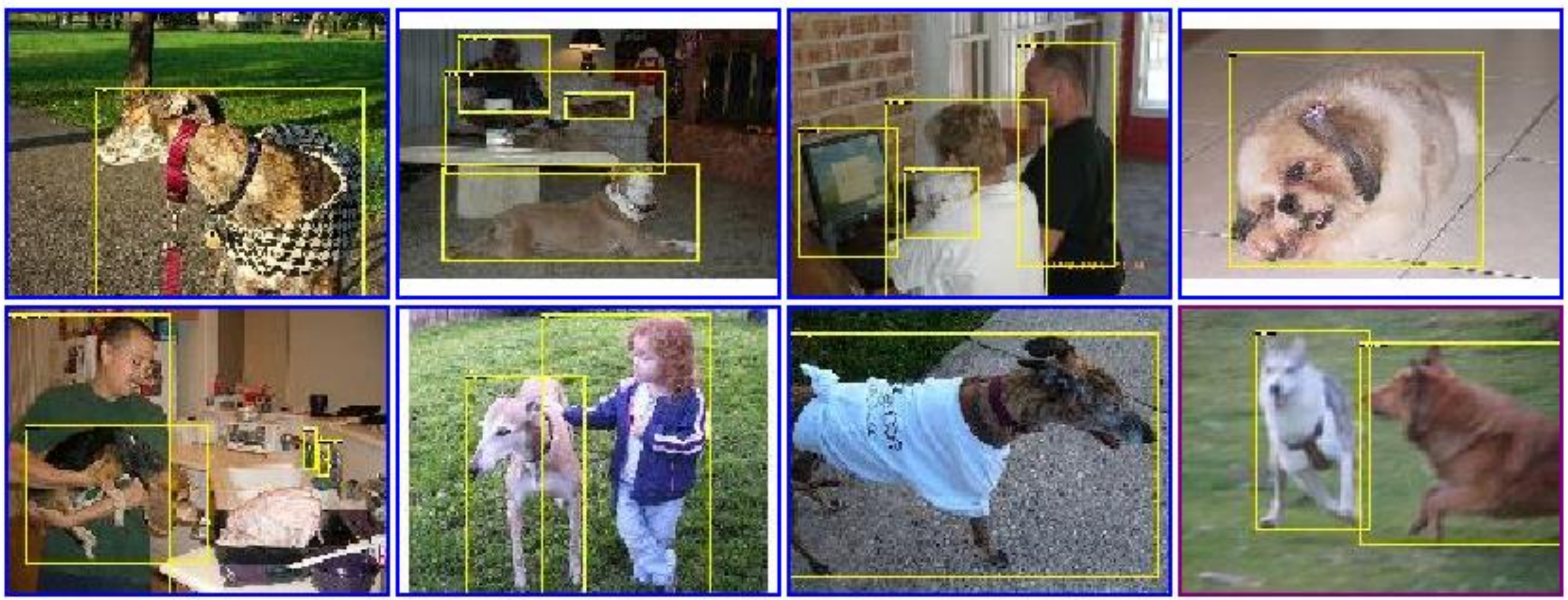
Limitations (continued)

- Not all objects are "box" shaped



Limitations (continued)

- Non-rigid, deformable objects not captured well with representations assuming a fixed 2D structure; or must assume fixed viewpoint
- Objects with less-regular textures not captured well with holistic appearance-based descriptions

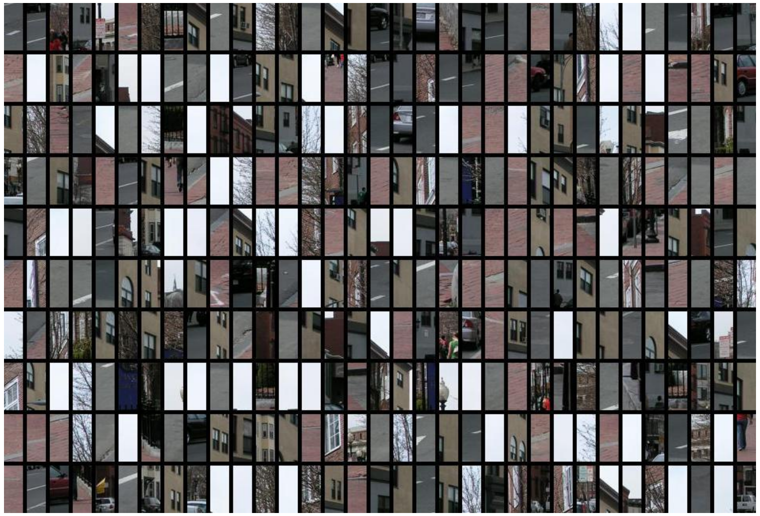


Slide credit: Kristen Grauman

Limitations (continued)

- If considering windows in isolation, context is lost

Sliding window



Detector's view

Limitations (continued)

- In practice, often entails large, cropped training set (expensive)
- Requiring good match to a global appearance description can lead to sensitivity to partial occlusions

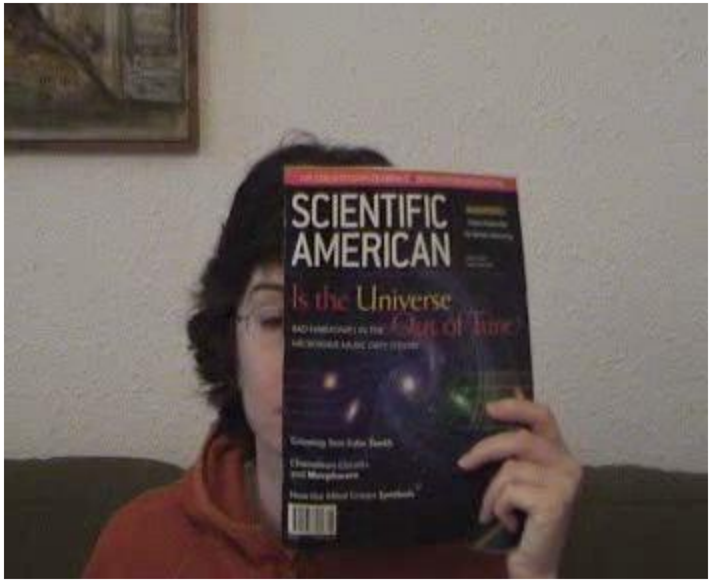


Image credit: Adam, Rivlin, & Shimshoni

References and Further Reading

- Read the HOG paper
  - N. Dalal, B. Triggs,
    Histograms of Oriented Gradients for Human Detection,
    CVPR, 2005.

- HOG Detector
  - Code available: http://pascal.inrialpes.fr/soft/olt/