SAIR

Spatial AI & Robotics Lab

CSE 473/573-A

L20: FACE DETECTION

Chen Wang

Spatial AI & Robotics Lab

Department of Computer Science and Engineering

University at Buffalo The State University of New York

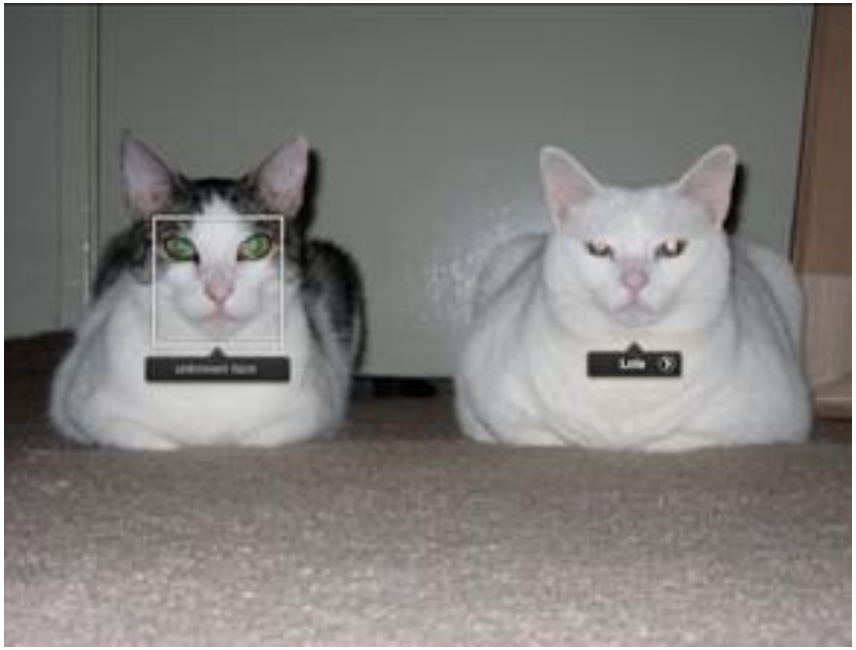
Many Slides from Lana Lazebnik

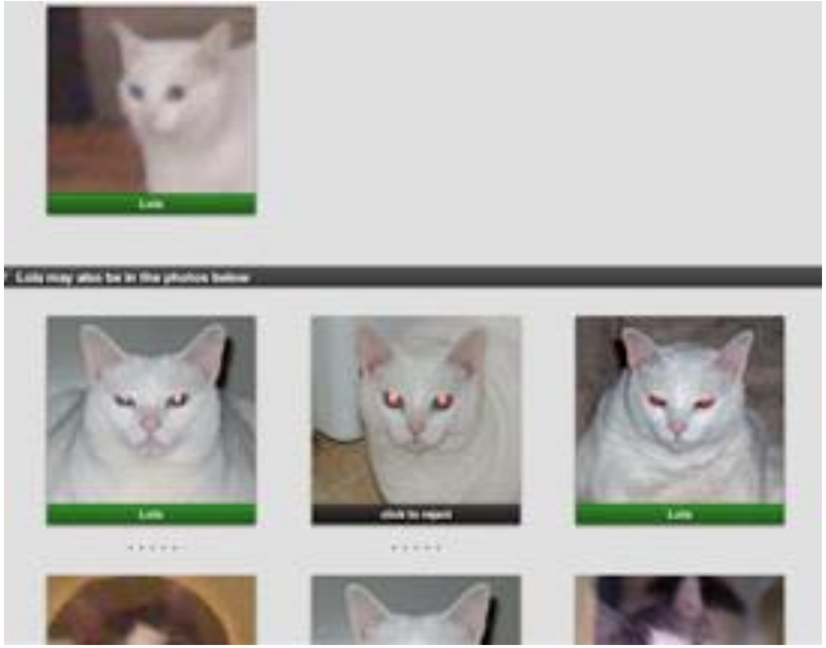
Face detection and recognition

Consumer application:

Consumer application:

- Can be trained to recognize pets!





Challenges of face detection

- Sliding window detector must evaluate tens of thousands of location/scale combinations
- Faces are rare: 0–10 per image
  - For efficiency, we should try to spend as little time as possible on the non-face windows
  - A megapixel image has ~10^6 pixels and a comparable number of candidate face locations
  - To avoid having a false positive in every image, our false positive rate has to be less than $1/10^6$

The Viola/Jones Face Detector

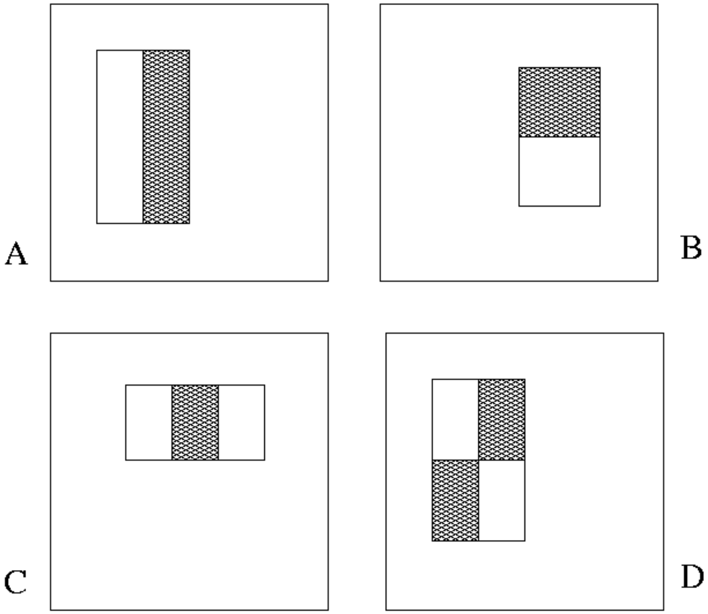
- Viola-Jones Detection Framework
  - Sliding Window Face Detection
- A seminal approach to real-time object detection
- Training is slow, but detection is very fast
- Key ideas
  - *Integral images* for fast feature evaluation
  - *Boosting* for feature selection
  - *Attentional cascade* for fast rejection of non-face patches

P. Viola and M. Jones. [Rapid object detection using a boosted cascade of simple features.] CVPR 2001.
P. Viola and M. Jones. [Robust real-time face detection.] IJCV 57(2), 2004.

Image Features

- Simple Features that measures the difference in intensity

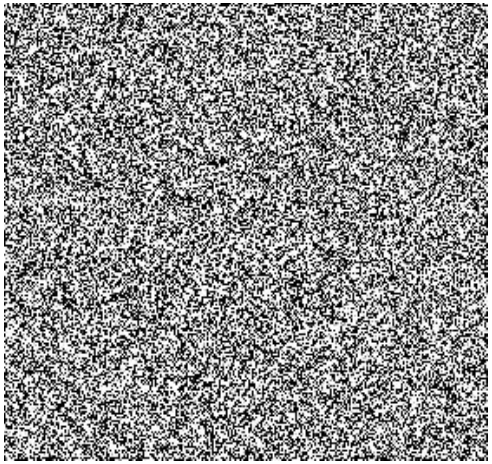
“Rectangle filters”



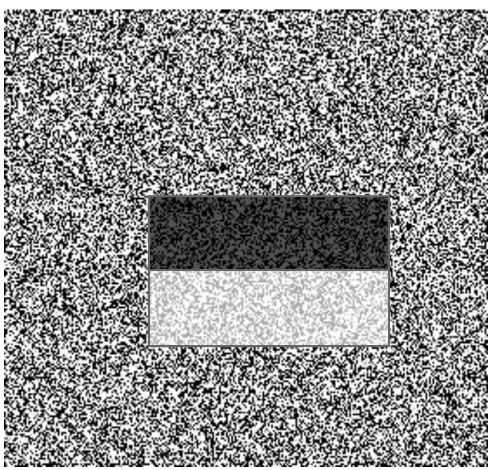
Value = ∑ (pixels in white) – ∑ (pixels in black)

Example

Source

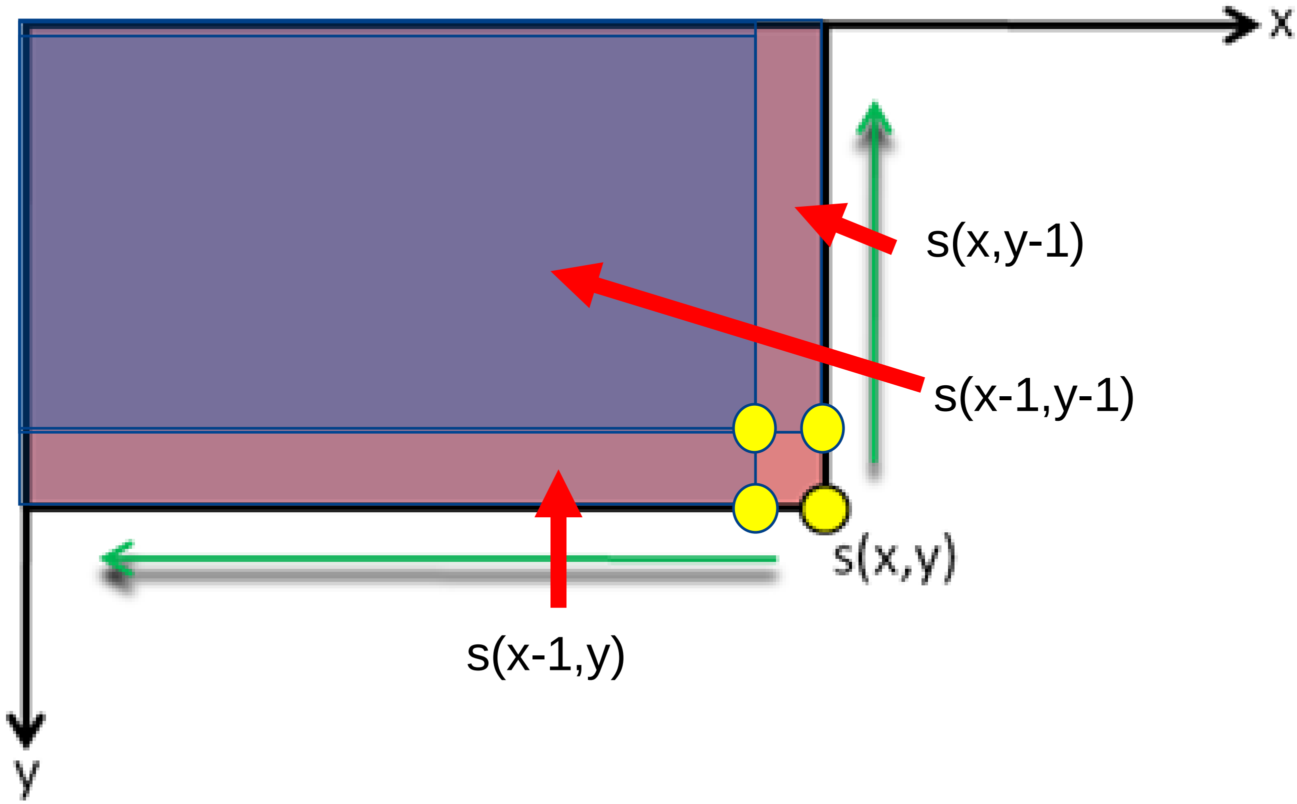


Result



Integral Image (Recap)

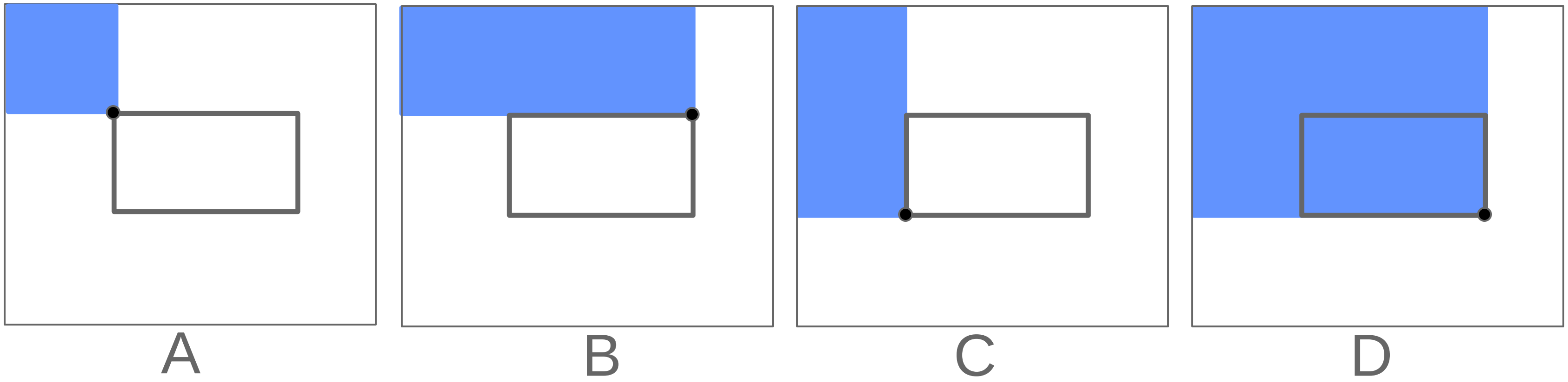
- A transformed image where every pixel is the sum of all pixels **above** and to the **left** of original image.



SAIR
Spatial AI & Robotics Lab

Integral images (Recap)

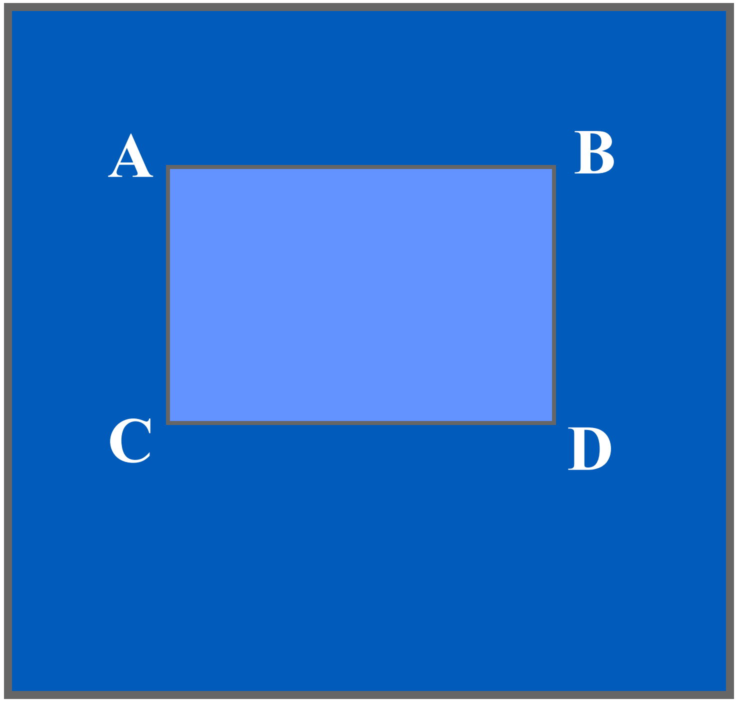
- What's the sum of pixels in the Rectangle ABCD?



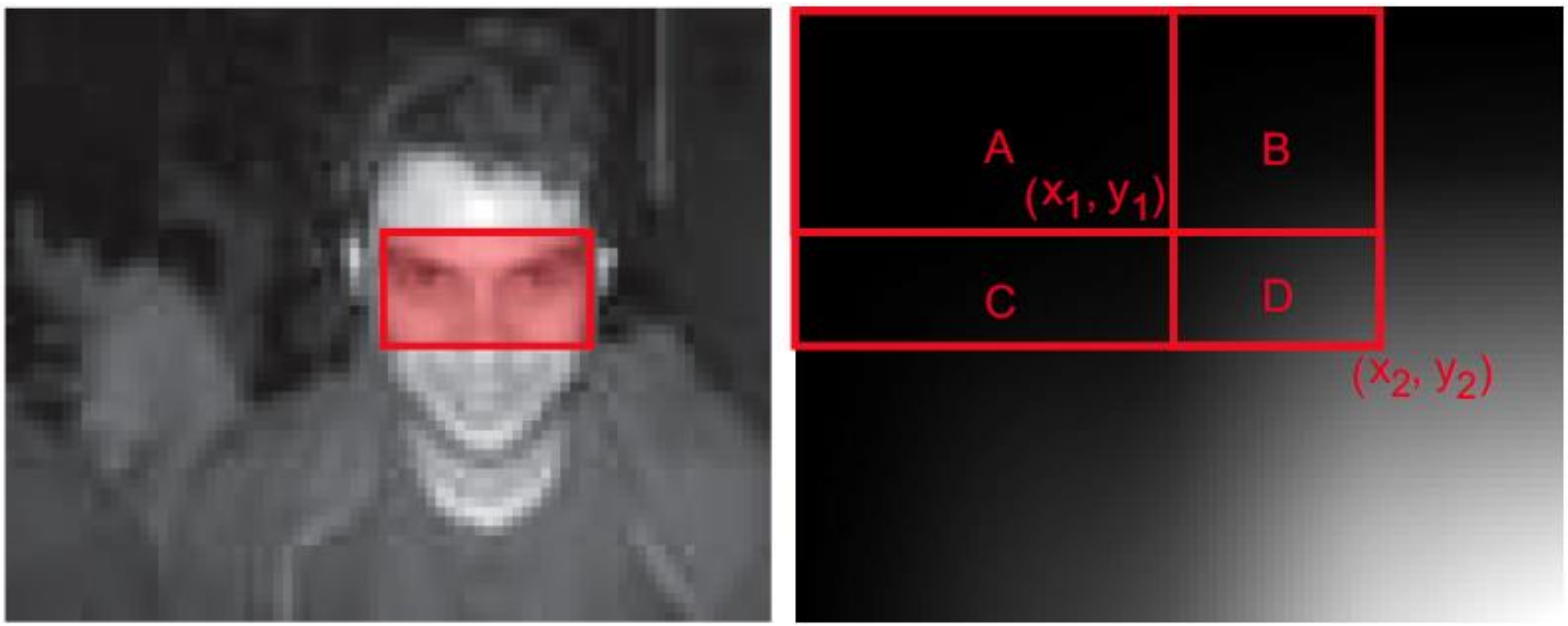
Computing sum within a rectangle (Recap)

- Let A,B,C,D be the values of the integral image at the corners of a rectangle
- Then the sum of original image values within the rectangle can be computed as:

  sum = D – B – C + A
- Only 3 additions are required for any size of rectangle!

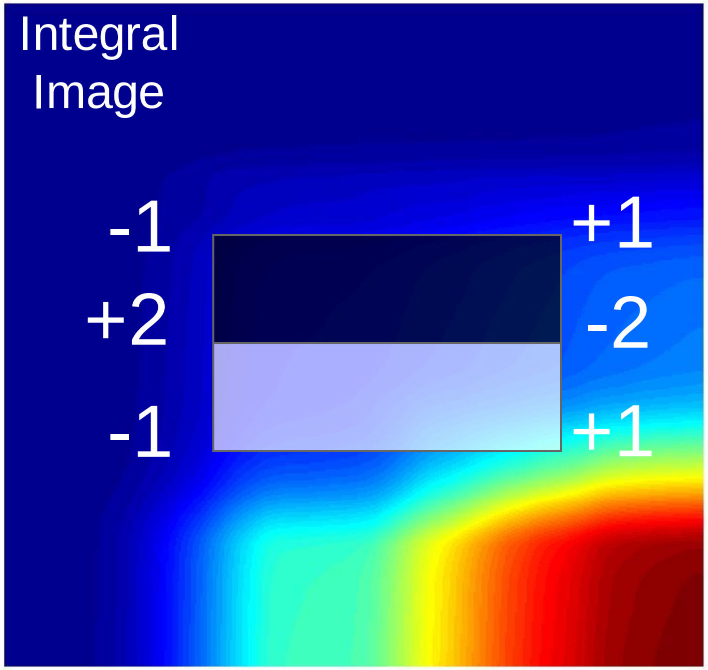


Integral Image Example (Recap)



Computing a rectangle feature

- Verify this:
  - Value = sum(scale factors * area of the regions).



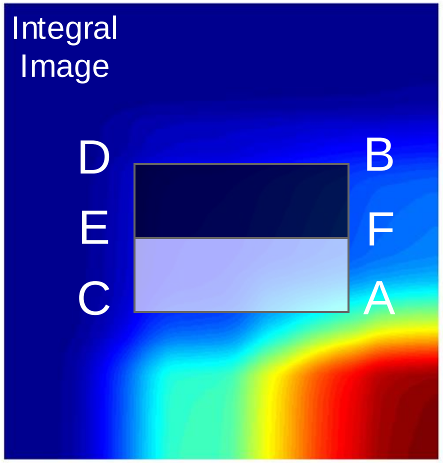
Exercise
10 minutes

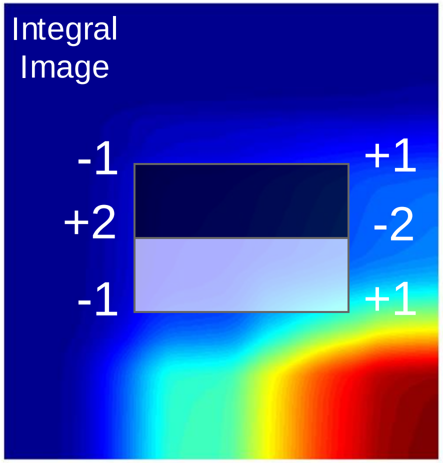
How do we get this?

- White Area – Black Area

- Sum of Entire Block = A – B – C + D
- White Block = A – F – C + E
- Black Block = F – E – B + D

- White – Black =
  - A – 2F – C +2E - D + B



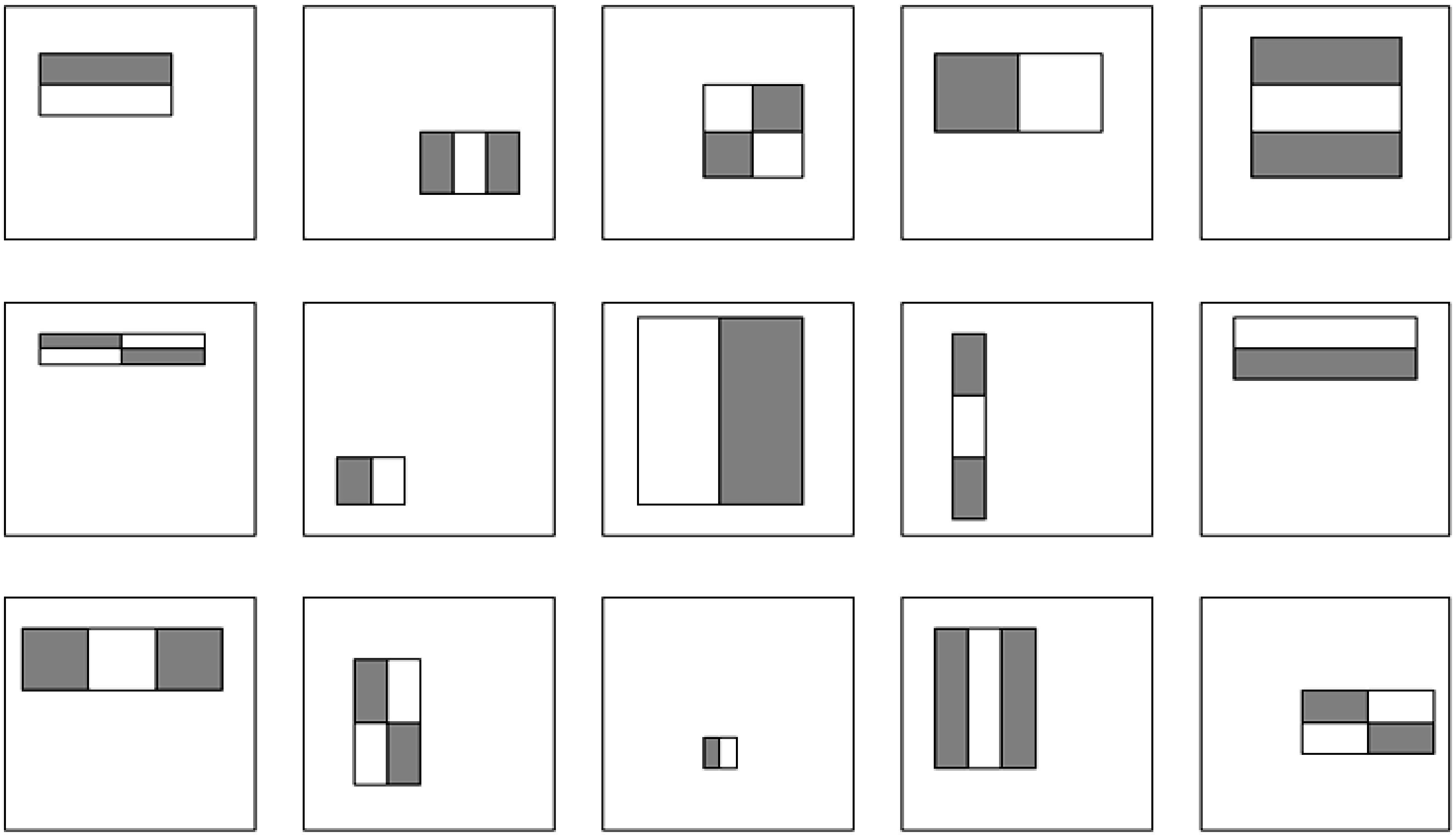


Key ideas

- *Integral images* for **fast feature evaluation**
- ***Boosting*** for feature selection
- *Attentional cascade* for fast rejection of non-face windows

Feature selection

- For a 24x24 detection region, the number of possible rectangle features is ~160,000!



SAIR
Spatial AI & Robotics Lab

Feature selection

- At test time, it is impractical to evaluate the entire feature set
- Can we create a good classifier using just a small subset of all possible features?
- How to select such a subset?

Boosting

- *Boosting* is a learning scheme that combines *weak learners* into a more accurate *ensemble classifier*
- Weak learners based on rectangle filters:

value of rectangle feature

$$h_t(x) = \begin{cases} 1 & \text{if } \; p_t f_t(x) > p_t \theta_t \\ 0 & \text{otherwise} \end{cases}$$

window

parity

threshold

- Ensemble classification function:

$$C(x) = \begin{cases} 1 & \text{if } \; \sum_{t=1}^{T} \alpha_t h_t(x) > \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

learned weights

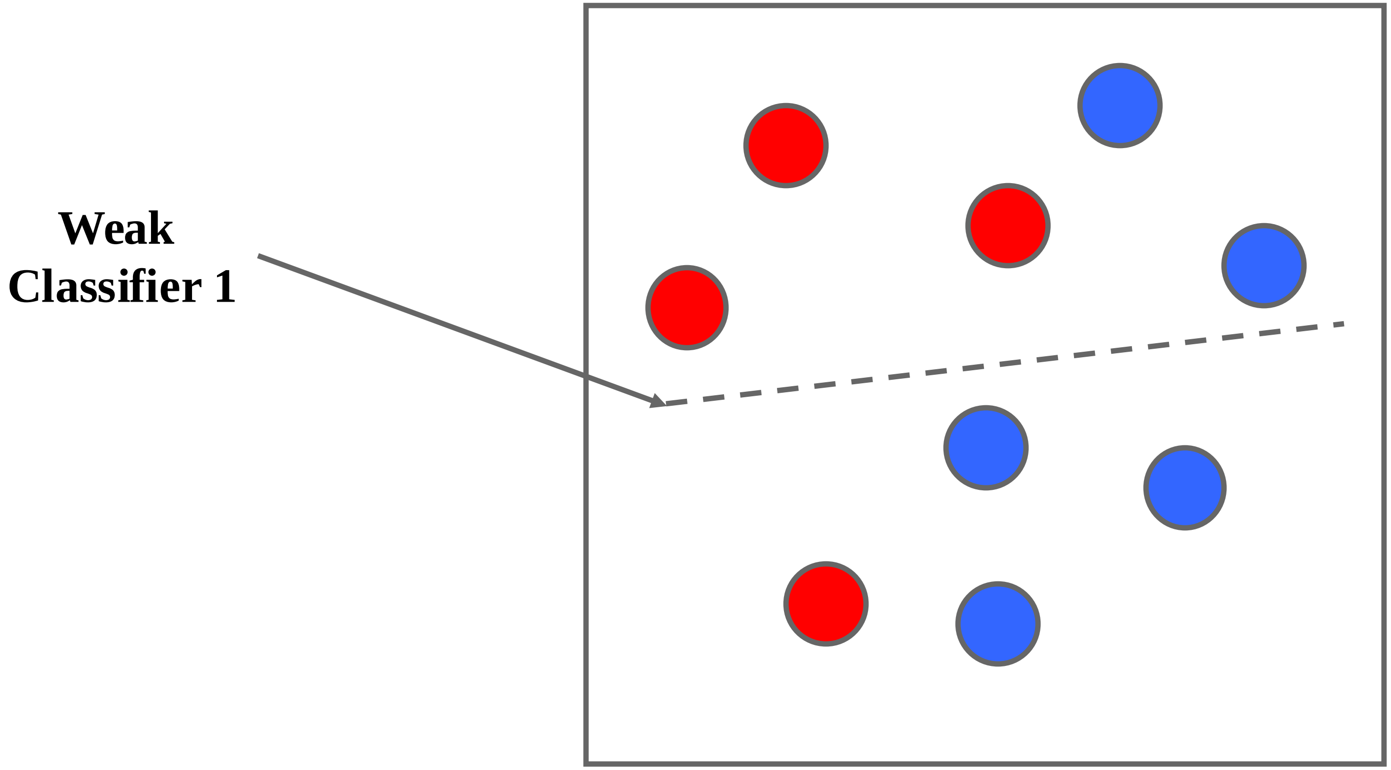
A parity indicates the direction of the inequality sign

Training procedure

- Initially, weight each training example equally
- In each boosting round:
  - Find the weak learner with lowest *weighted* training error
  - Raise the weights of training examples misclassified by current weak learner
- Compute final classifier as linear combination of all weak learners
  - Weight of each learner is proportional to its accuracy
  - Exact formulas for re-weighting and combining weak learners depend on the particular boosting scheme, i,e. Adaptive Boost (AdaBoost).

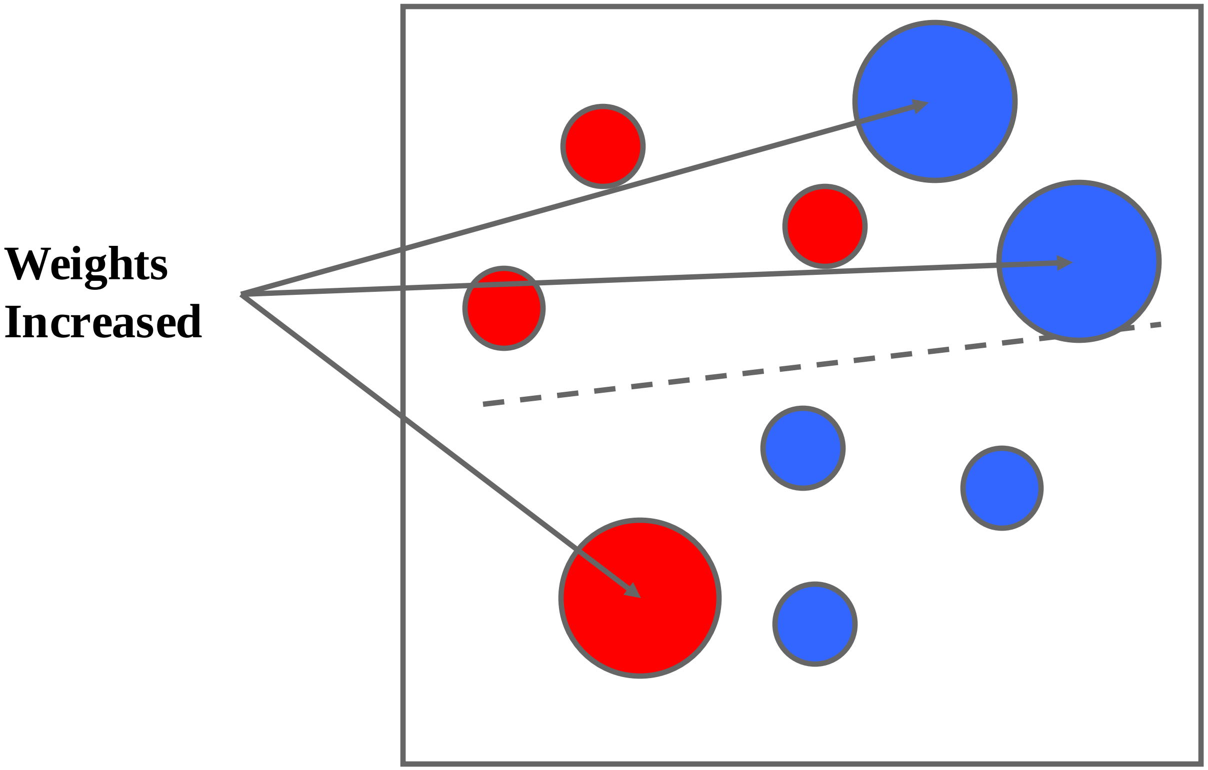
Y. Freund and R. Schapire, A short introduction to boosting, *Journal of Japanese Society for Artificial Intelligence*, 14(5):771-780, September, 1999.

Boosting intuition

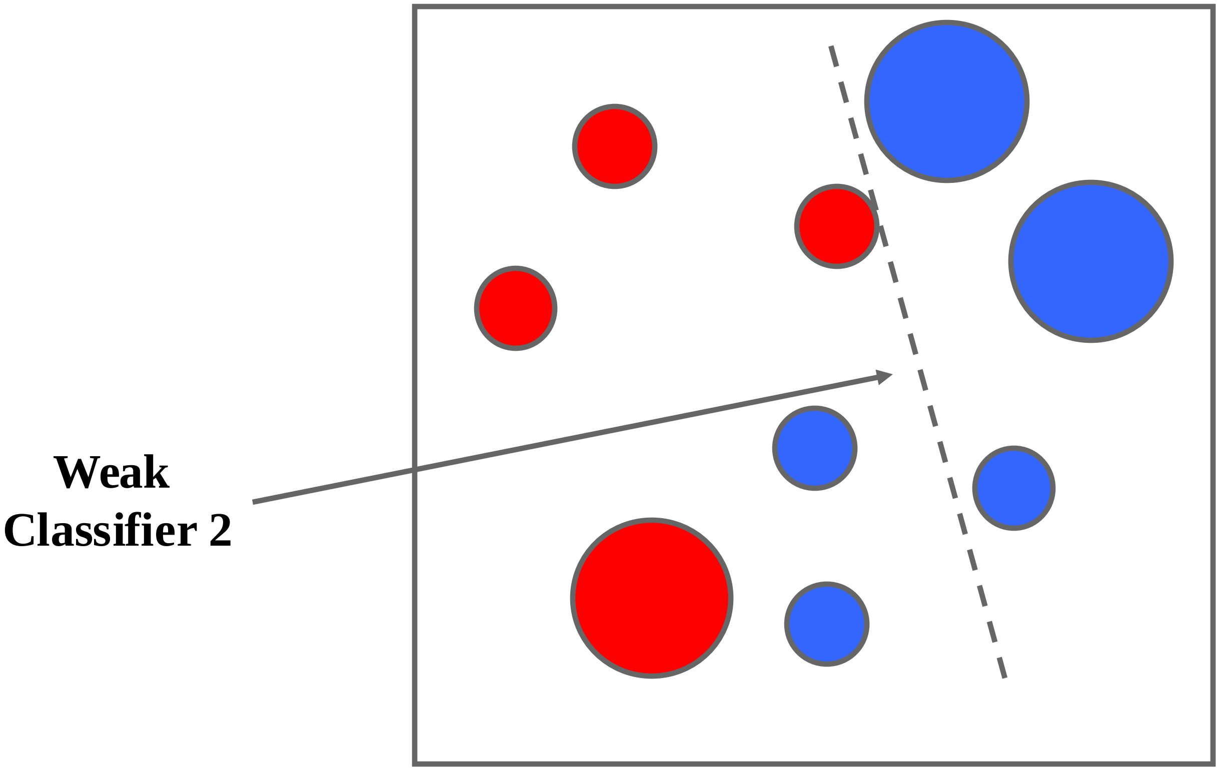


Slide credit: Paul Viola

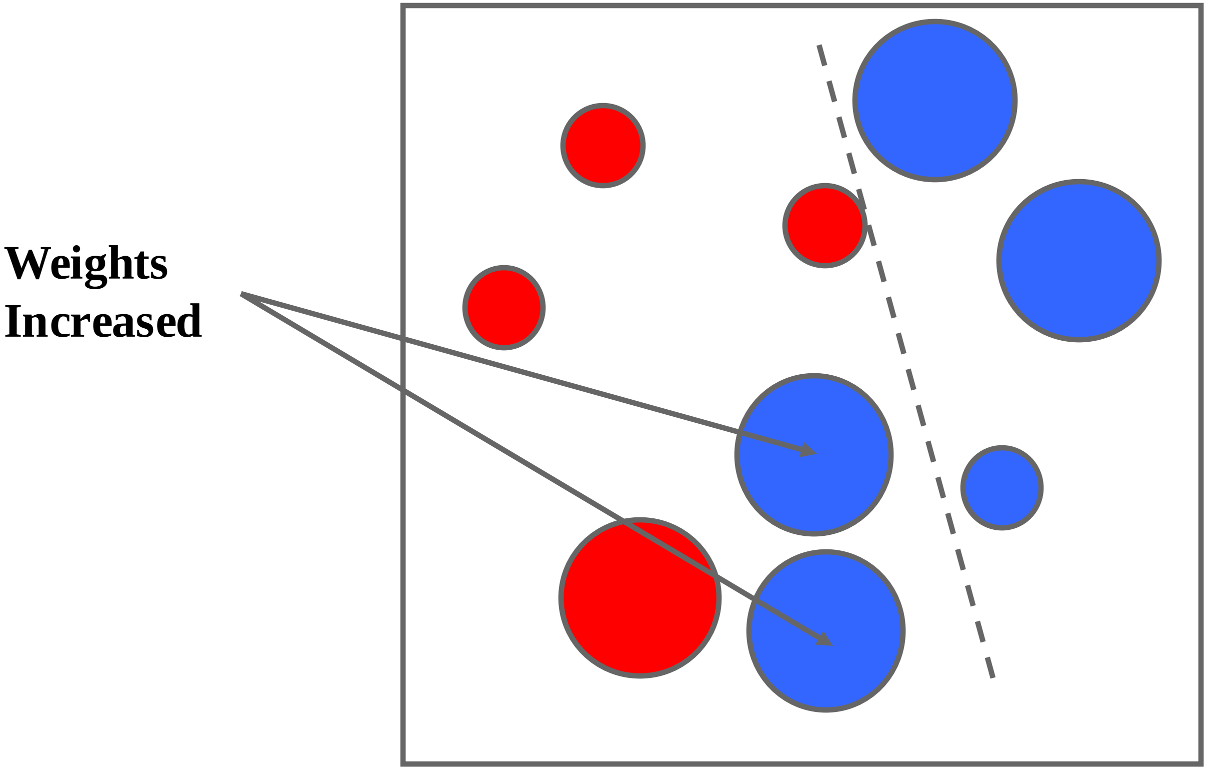
Boosting illustration



Boosting illustration

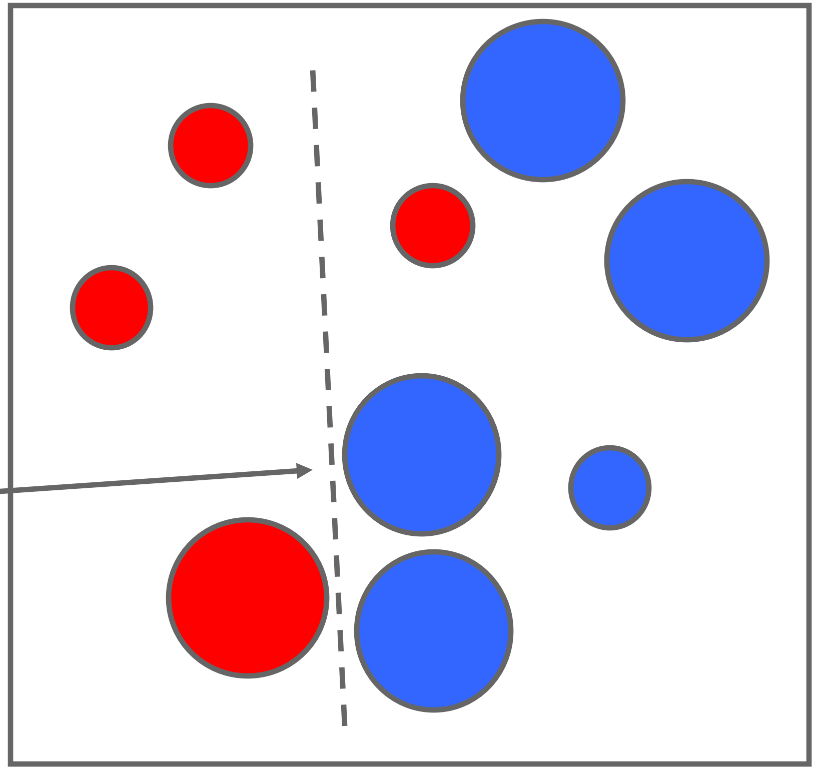


Boosting illustration



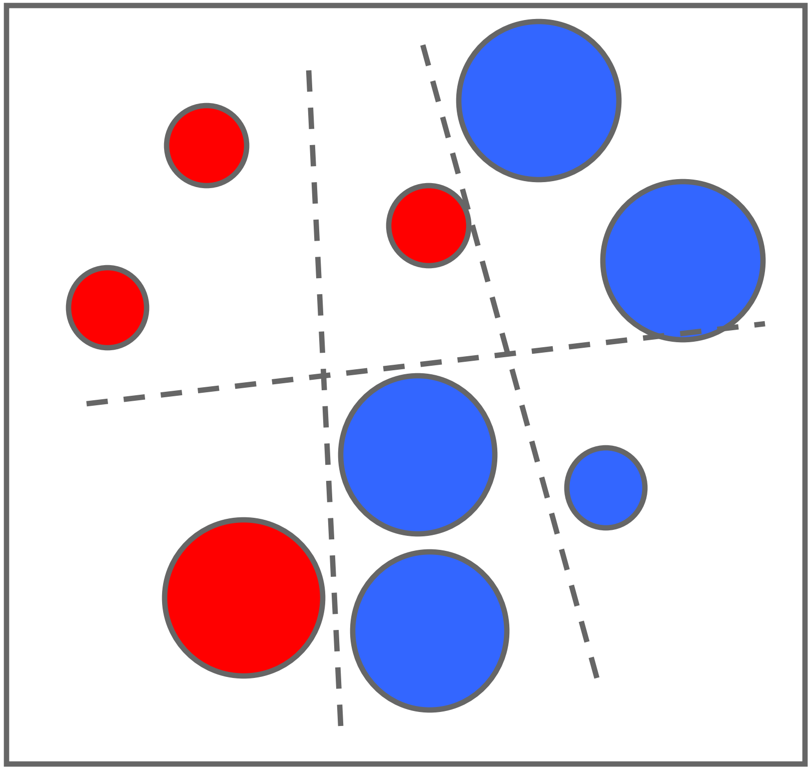
Boosting illustration

Weak
Classifier 3



Boosting illustration

Final classifier is a combination of weak classifiers



Training procedure

- Initially, weight each training example equally
- In each boosting round:
  - Find the weak learner with lowest *weighted* training error
  - Raise the weights of training examples misclassified by current weak learner
- Compute final classifier as linear combination of all weak learners
  - Weight of each learner is proportional to its accuracy
  - Exact formulas for re-weighting and combining weak learners depend on the particular boosting scheme, i,e. Adaptive Boost (AdaBoost)

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

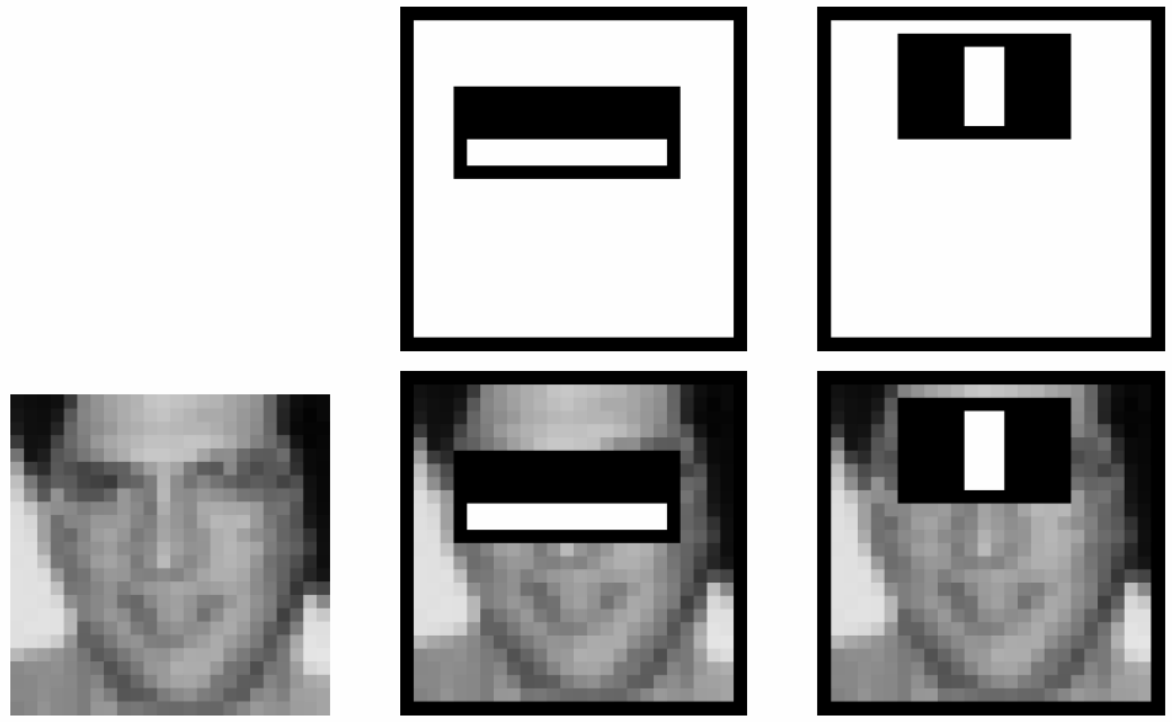
Adaboost for face detection

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:
  1. Normalize the weights,
  $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$
  so that w_t is a probability distribution.
  2. For each feature, j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t, $\epsilon_j = \sum_i w_i \left| h_j(x_i) - y_i \right|$.
  3. Choose the classifier, h_t, with the lowest error ϵ_t.
  4. Update the weights:
  $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$
  where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
- The final strong classifier is:
$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$
where $\alpha_t = \log \frac{1}{\beta_t}$

P. Viola and M. Jones. *Rapid object detection using a boosted cascade of simple features.* CVPR 2001.

Boosting for face detection

- First two features selected by boosting:
- This feature combination can yield 100% recall and 50% false positive rate



Boosting vs. SVM

- Advantages of boosting
  - Integrates classifier training with feature selection
  - Complexity of training is linear instead of quadratic in the number of training examples
  - Flexibility in the choice of weak learners, boosting scheme
  - Testing is fast
- Disadvantages
  - Needs many training examples
  - Training is slow
  - Often doesn't work as well as SVM, especially for many-class problems

Key ideas

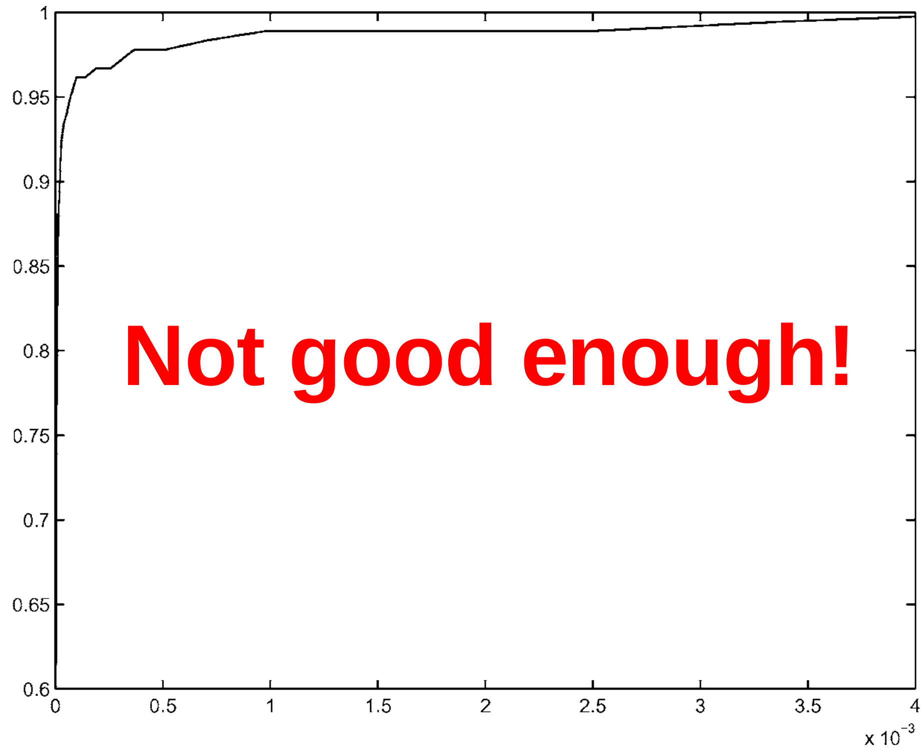
- *Integral images* for **fast feature evaluation**
- *Boosting* for feature selection
- *Attentional* ***cascade*** for fast rejection of non-face windows

Boosting for face detection

- A 200-feature classifier can yield 95% detection rate and a false positive rate of 1/14084

ROC curve

Recall

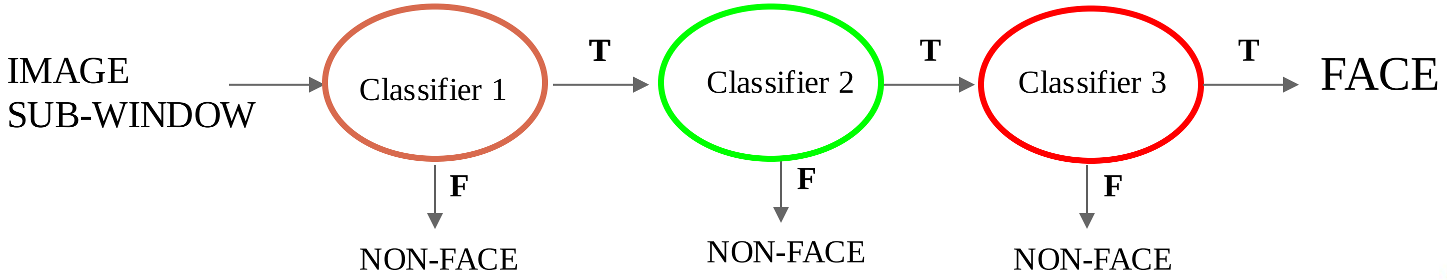


Not good enough!

False positive rate

Attentional cascade

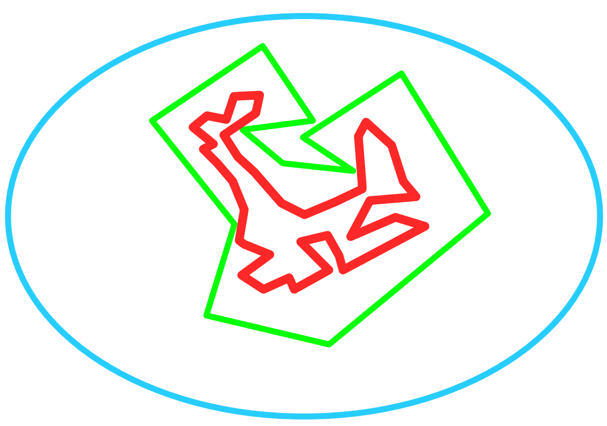
- We start with simple classifiers which reject many of the negative sub-windows while detecting almost all positive sub-windows
- Positive response from the first classifier triggers the evaluation of a second (more complex) classifier, etc.
- A negative outcome at any point leads to the immediate rejection of the sub-window



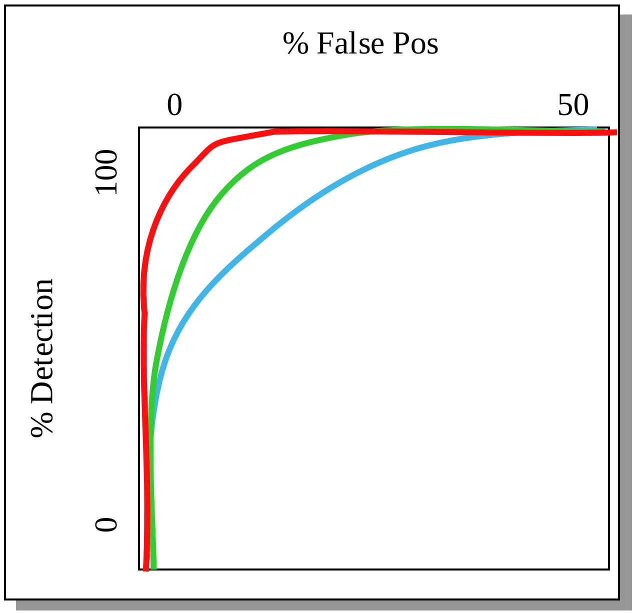
SAIR
Spatial AI & Robotics Lab

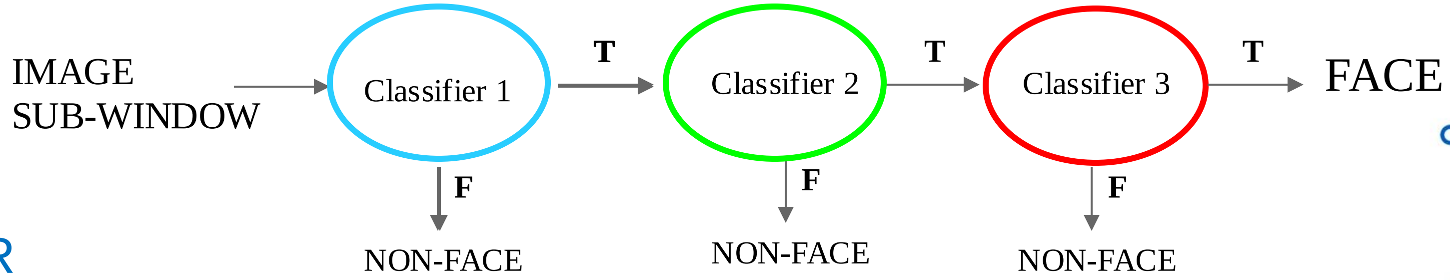
Attentional cascade

- Chain classifiers that are progressively more complex and have lower false positive rates



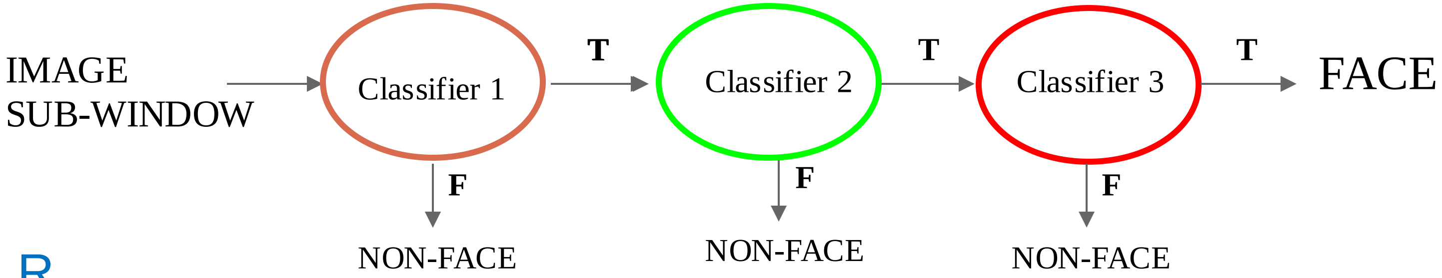
ROC





Attentional cascade

- The detection rate and the false positive rate of the cascade are found by multiplying the respective rates of the individual stages
- A detection rate of 0.9 and a false positive rate on the order of 10^{-6} can be achieved by a 10-stage cascade if each stage has a detection rate of 0.99 ($0.99^{10} \approx 0.9$) and a false positive rate of about 0.30 ($0.3^{10} \approx 6\times10^{-6}$)



Training the cascade

- Set target detection and false positive rates for each stage
- Keep adding features to current stage until target rates met
  - Lower AdaBoost threshold to maximize detection
    - opposed to minimizing total classification error
  - Test on a *validation set*
- If the overall false positive rate is not low enough
  - Add another stage
- Use false positives from current stage as the negative training examples for the next stage

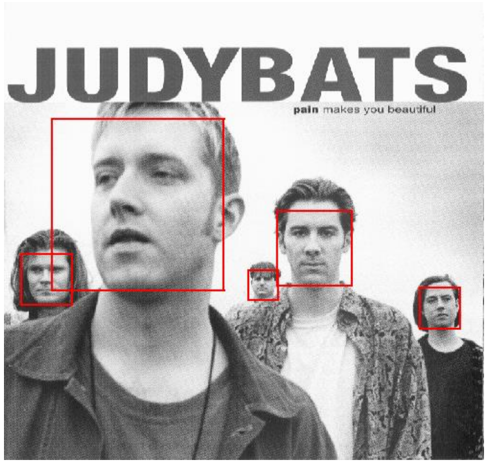
The implemented system

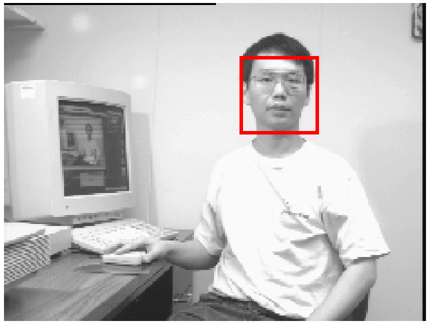
- Training Data
  - 5000 faces
    - All frontal, rescaled to 24x24 pixels
  - 300 million non-faces
    - 9500 non-face images
  - Faces are normalized
    - Scale, translation
- Many variations
  - Across individuals
  - Illumination
  - Pose

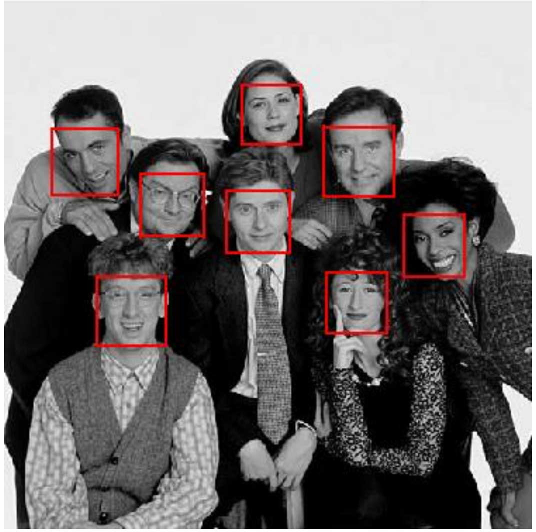
System performance

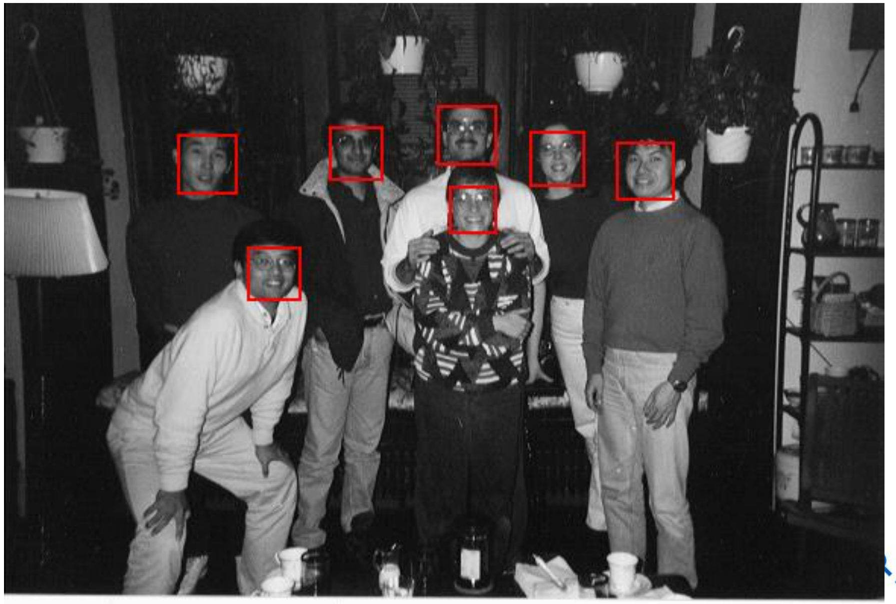
- Training time: “weeks” on 466 MHz Sun workstation
- 38 layers, total of 6061 features
- Average of 10 features evaluated per window on test set
- “On a 700 Mhz Pentium III processor, the face detector can process a 384 by 288 pixel image in about .067 seconds”
  - 15 Hz
  - 15 times faster than previous detector of comparable accuracy (Rowley et al.,1998)

Output of Face Detector on Test Images



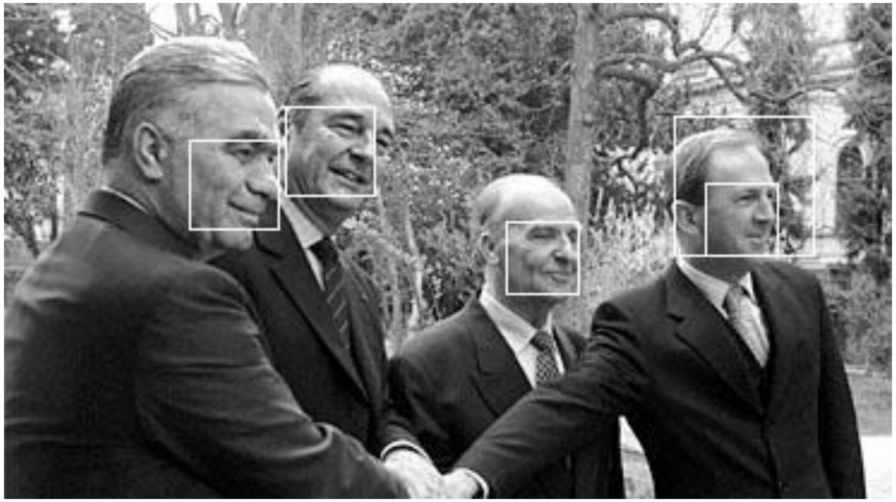






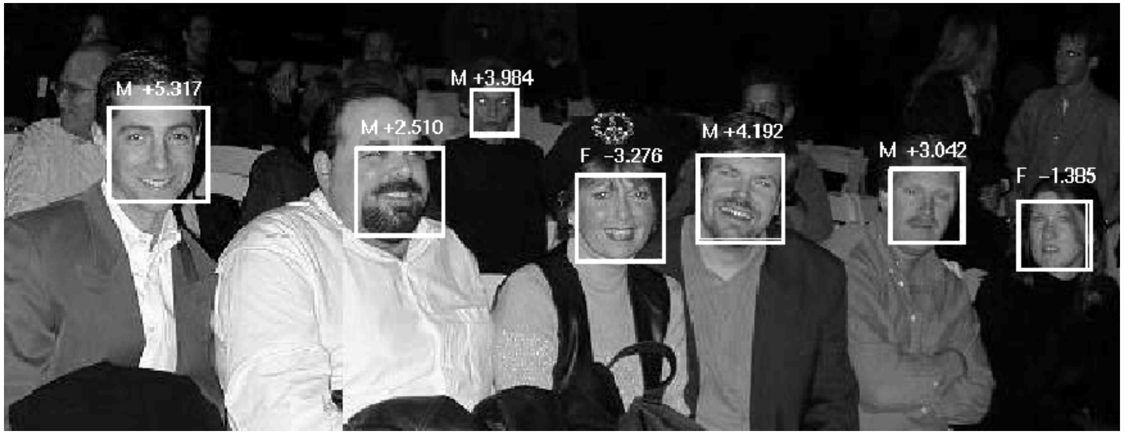
Other detection tasks

Facial Feature Localization

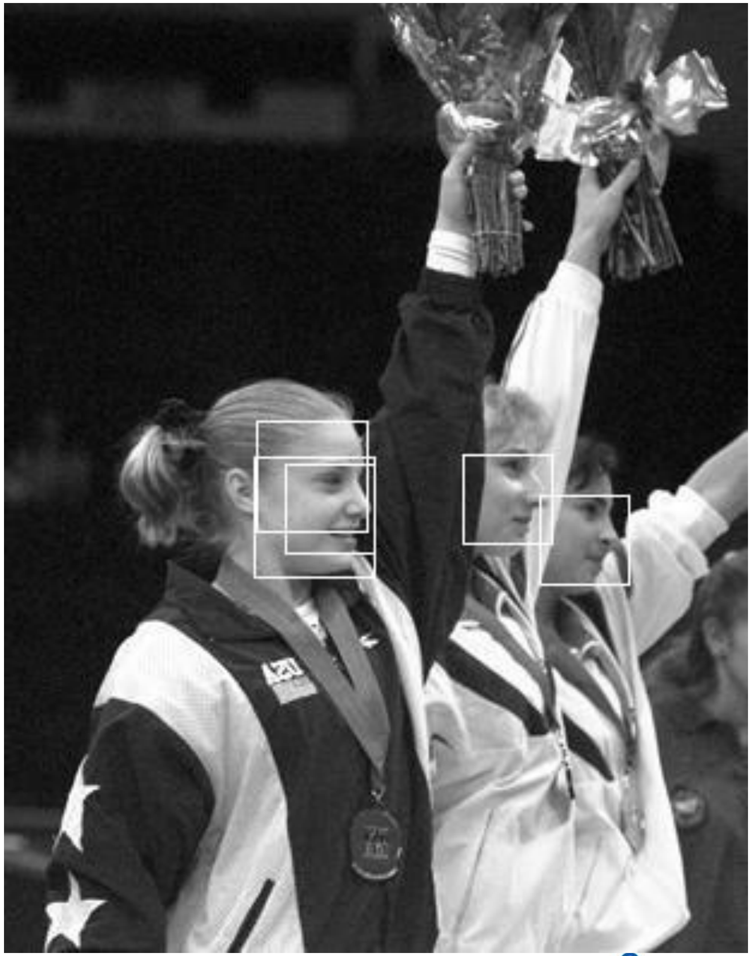


Profile Detection

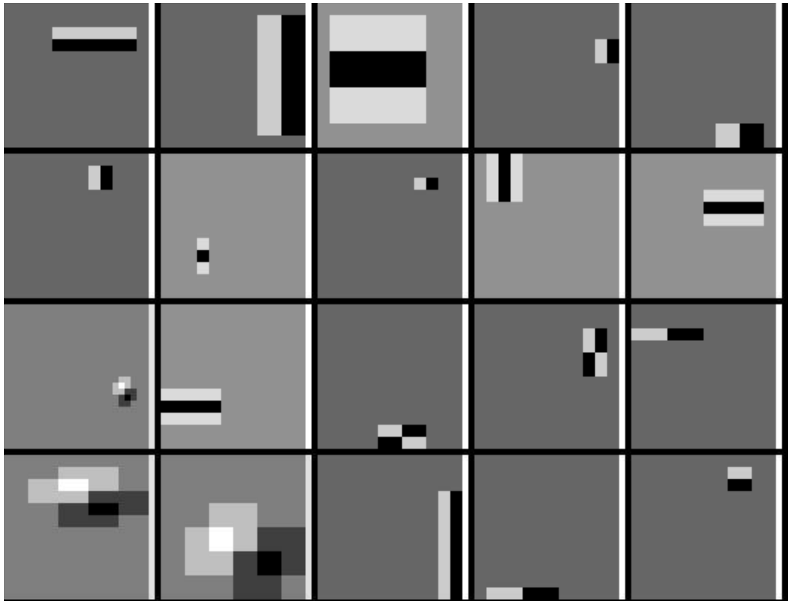
Male vs. female



Profile Detection



Profile Features



Summary: Viola/Jones detector

- Rectangle features
- Integral images for fast computation
- Boosting for feature selection
- Attentional cascade for fast rejection of negative windows