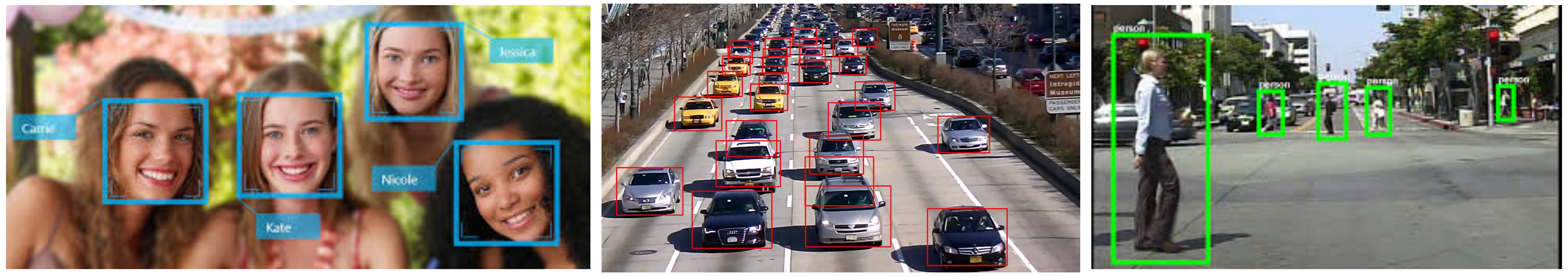
CPSC 425: Computer Vision



Lecture 22: Object Detection

Object Detection: Introduction

We have been discussing **image classification**, where we pass a whole image into a classifier and obtain a class label as output

We assumed the image contained a single, central object

The task of **object detection** is to detect and localize all instances of a target object class in an image
— Localization typically means putting a tight bounding box around the object

Sliding Window

Train an image classifier as described previously. ‘Slide’ a fixed-sized detection window across the image and evaluate the classifier on each window.

Image credit: KITTI Vision Benchmark

Sliding Window

Train an image classifier as described previously. ‘Slide’ a fixed-sized detection window across the image and evaluate the classifier on each window.

Image credit: KITTI Vision Benchmark

This is a search over location
— We have to search over scale as well
— We may also have to search over aspect ratios

Example: Face Detection

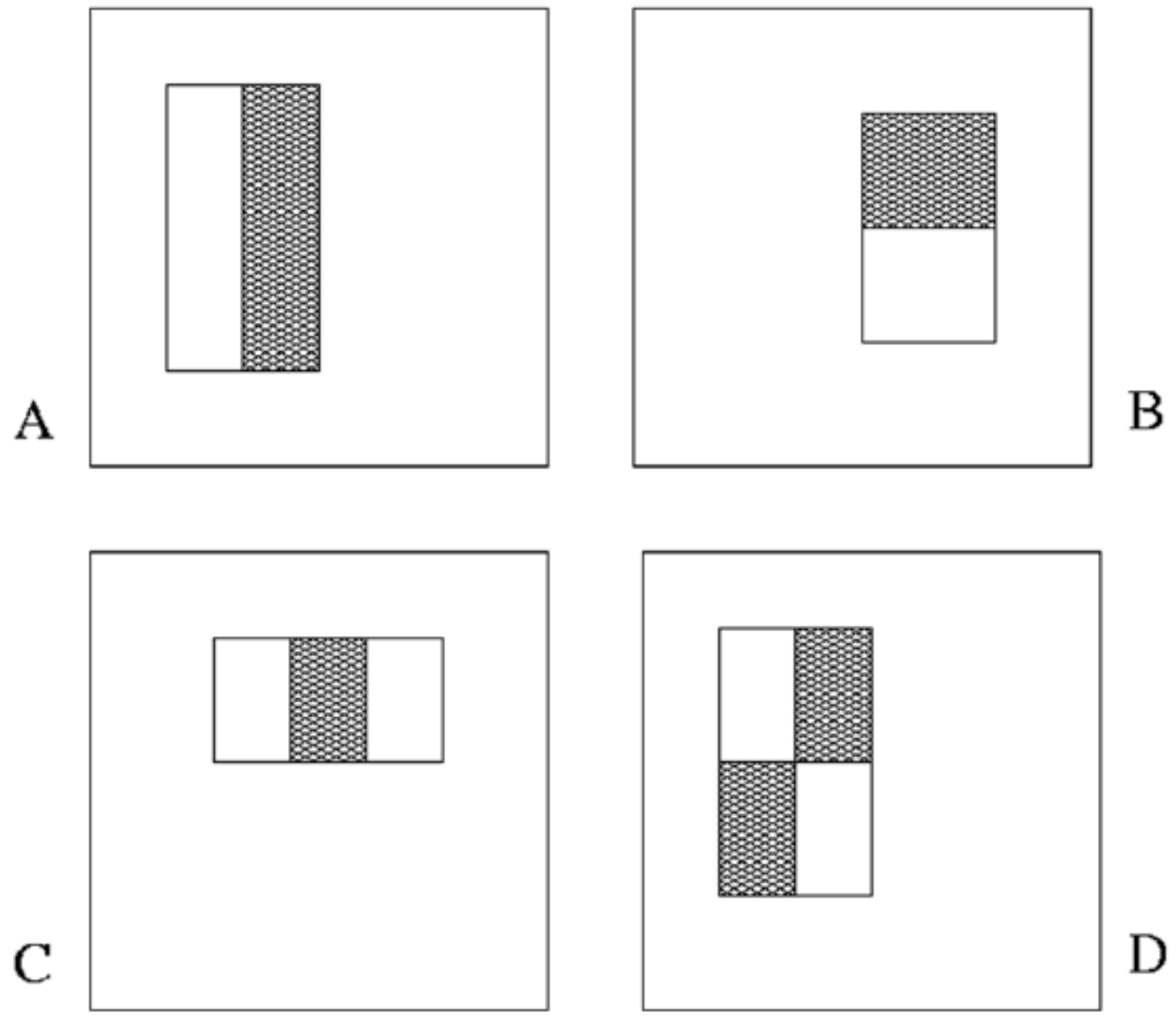
The **Viola-Jones** face detector is a classic sliding window detector that learns both efficient features and a classifier

A key strategy is to use features that are fast to evaluate to reject most windows early

The Viola-Jones detector computes ‘rectangular’ features within each window

Example: Face Detection

A ‘rectangular’ feature is computed by summing up pixel values within rectangular regions and then differencing those region sums



a.k.a. **Harr** Wavelets

Figure credit: P. Viola and M. Jones, 2001

Intergral Image

$$A(x, y) = \sum_{x' \leq x, y' \leq y} I(x', y')$$

$I(x, y)$

1	5	2
2	4	1
2	1	1

original image

$A(x, y)$

1	6	8
3	12	15
5	15	19

integral image

Image Credit: Ioannis (Yannis) Gkioulekas (CMU)

Intergral Image

What is the sum of the bottom right 2x2 square?

$I(x, y)$

1	5	2
2	4	1
2	1	1

original image

$A(x, y)$

1	6	8
3	12	15
5	15	19

integral image

Image Credit: Ioannis (Yannis) Gkioulekas (CMU)

Intergral Image

What is the sum of the bottom right 2x2 square?

$I(x, y)$

1	5	2
2	4	1
2	1	1

original image

$A(x, y)$

1	6	8
3	12	15
5	15	19

integral image

Image Credit: Ioannis (Yannis) Gkioulekas (CMU)

Intergral Image

What is the sum of the bottom right 2x2 square?

$I(x, y)$

1	5	2
2	4	1
2	1	1

original image

$A(x, y)$

1	6	8
3	12	15
5	15	19

integral image

$$
\begin{aligned}
A(1,1,3,3) &= A(3,3) - A(1,3) - A(3,1) + A(1,1) \\
&= \quad 19 \quad - \quad 8 \quad - \quad 5 \quad + \quad 1 \\
&= \quad 7
\end{aligned}
$$

Image Credit: Ioannis (Yannis) Gkioulekas (CMU)

Intergral Image

$$A(x,y) = \sum_{x' \le x, y' \le y} I(x', y')$$

$I(x,y)$

1	5	2
2	4	1
2	1	1

original image

$A(x,y)$

1	6	8
3	12	15
5	15	19

integral image

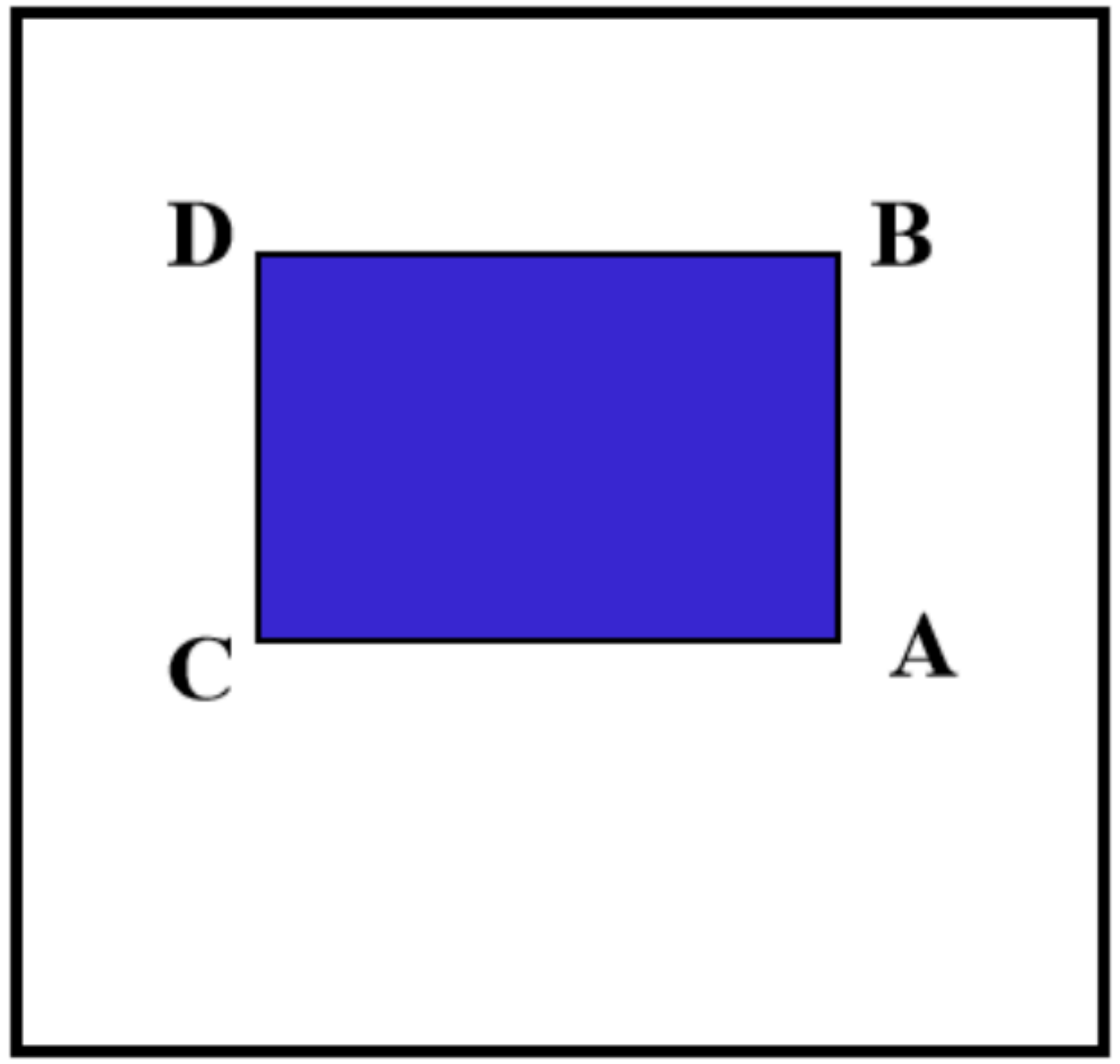
Can find the **sum** of any block using **3** operations

$$A(x_1, y_1, x_2, y_2) = A(x_2, y_2) - A(x_1, y_2) - A(x_2, y_1) + A(x_1, y_1)$$

Image Credit: Ioannis (Yannis) Gkioulekas (CMU)

Example: Face Detection

Given an integral image, the sum within a rectangular region in I can be computed with just 3 additions



Sum = A - B - C + D

Figure credit: P. Viola

Constant time: does not depend on the size of the region. We can avoid scaling images - just scale features directly (remember template matching!)

Integral Image Layer for Deep Neural Networks

In a classical paper [1] from 2001, Viola and Jones popularized the use of large rectangular image filters in order to obtain features for image recognition. The use of very large filters allowed Viola and Jones to compute features over very large receptive fields without blowing up the computation cost. For the next 10+ years, such features remained the staple of fast computer vision (e.g. [2]). The advent of deep learning made the use of integral-image features far less popular. Currently, state-of-the-art architectures invariably relying on very deep architectures. In these architectures sufficiently large receptive fields are obtained via the use of downsampling with subsequent upsampling [3] or via dilated convolutions [4]. All such tricks however have their downsides and usually necessitate the use of very deep networks.

The goal of this project is to implement an integral image-based filtering as a layer for deep architectures in Torch deep learning package, and to evaluate it for the task of learning very fast object detectors (as an alternative to e.g. [5]) and semantic segmentation systems (as an alternative to e.g. [3,4]). The hope is to obtain much shallower architectures, which at least for simple classes (e.g. roadsigns or upright pedestrians) will approach the performance of much deeper ones.

The project is supervised by Victor Lempitsky at Skoltech, Moscow, Russia.

https://github.com/shrubb/integral-layer

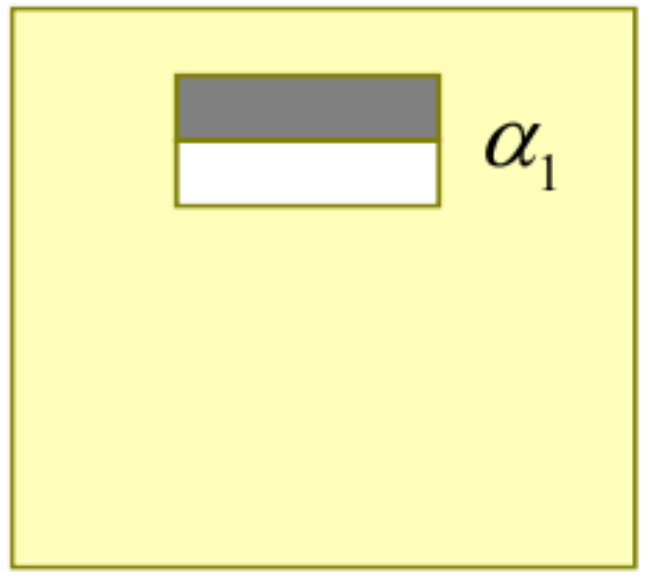
Deep Neural Networks for Object Detection

Christian Szegedy **Alexander Toshev** **Dumitru Erhan**
Google, Inc.
{szegedy, toshev, dumitru}@google.com

Abstract

Deep Neural Networks (DNNs) have recently shown outstanding performance on image classification tasks [14]. In this paper we go one step further and address the problem of object detection using DNNs, that is not only classifying but also precisely localizing objects of various classes. We present a simple and yet powerful formulation of object detection as a regression problem to object bounding box masks. We define a multi-scale inference procedure which is able to produce high-resolution object detections at a low cost by a few network applications. State-of-the-art performance of the approach is shown on Pascal VOC.

Example: Face Detection



$(x_1, 1)$ $(x_2, 1)$ $(x_3, 0)$ $(x_4, 0)$ $(x_5, 0)$ $(x_6, 0)$

..... (x_n, y_n)

0.8 0.7 0.2 0.3 0.8 0.1

Weak classifier $h_j(x) = \begin{cases} 1 & \text{if } f_j(x) > \theta_j \\ 0 & \text{otherwise} \end{cases}$

threshold

Image Credit: Ioannis (Yannis) Gkioulekas (CMU)

Example: Face Detection

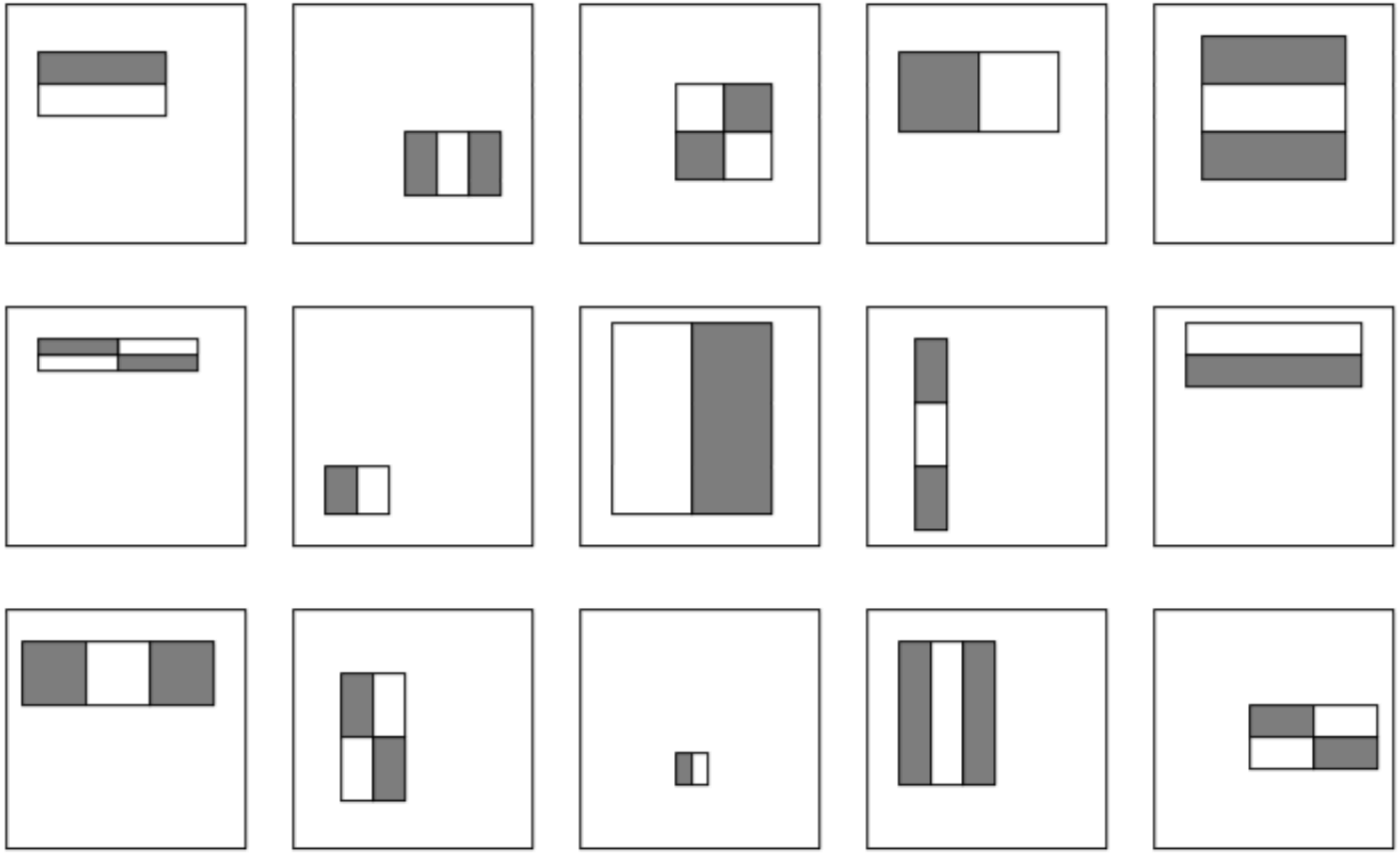


Figure credit: B. Freeman

Many possible rectangular features (180,000+ were used in the original paper)

Example: Face Detection

Use **boosting** to both select the informative features and form the classifier. Each round chooses a weak classifier that simply compares a single rectangular feature against a threshold

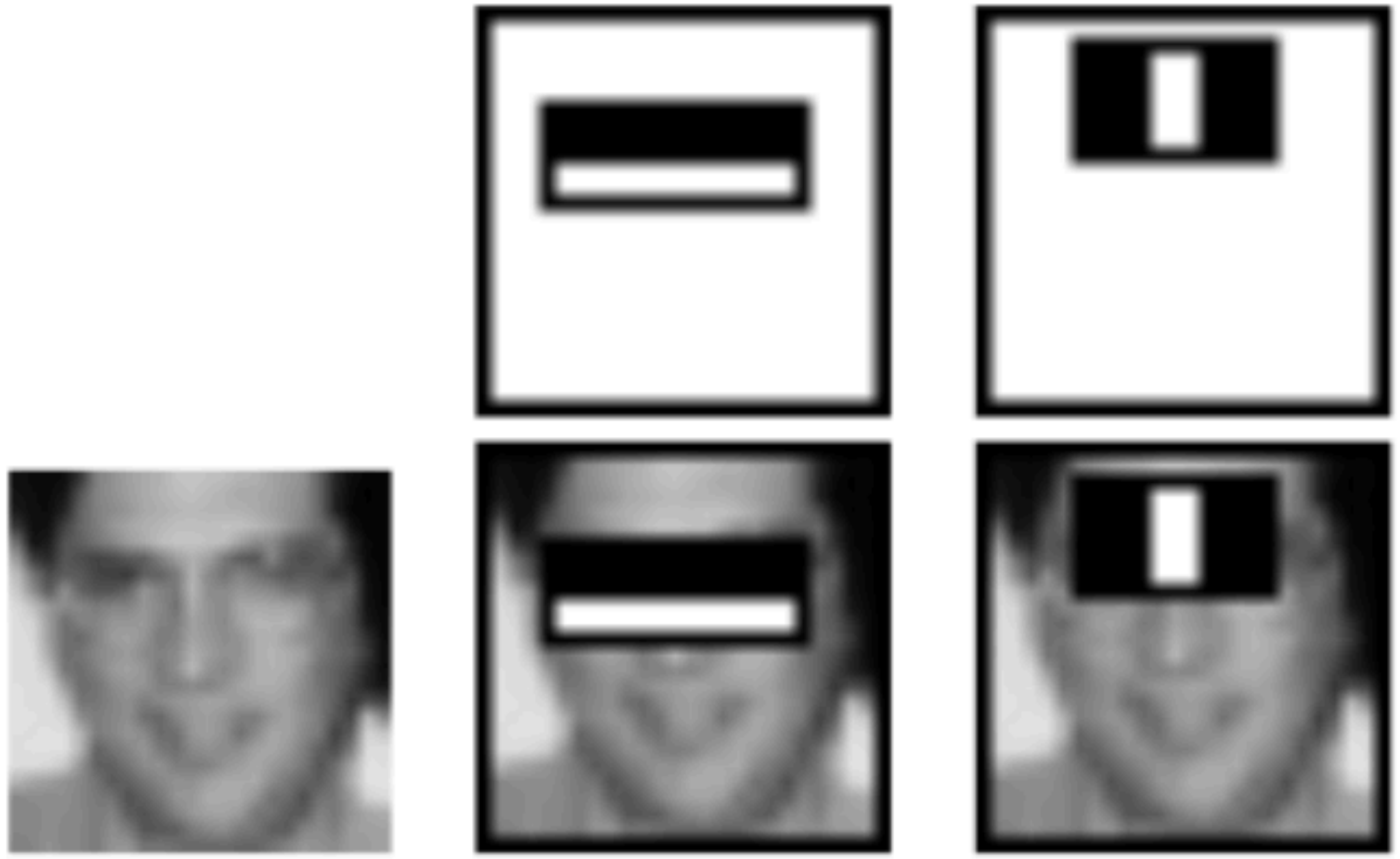


Figure credit: P. Viola and M. Jones, 2001

Example: Face Detection

2. Select best filter/threshold combination

a. Normalize the weights $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$

$$h_j(x) = \begin{cases} 1 & \text{if } f_j(x) > \theta_j \\ 0 & \text{otherwise} \end{cases}$$

b. For each feature, *j* $\varepsilon_j = \sum_i w_i \left| h_j(x_i) - y_i \right|$

c. Choose the classifier, h_t with the lowest error ε_t

3. Reweight examples

$$w_{t+1,i} = w_{t,i} \beta_t^{1-|h_t(x_i)-y_i|} \qquad \beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$$

Image Credit: Ioannis (Yannis) Gkioulekas (CMU)

Cascading Classifiers

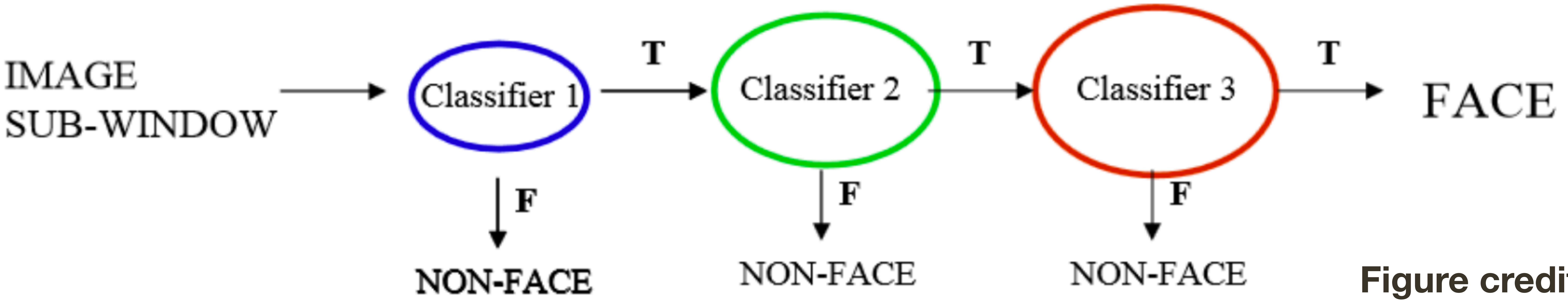


Figure credit: P. Viola

To make detection **faster**, features can be reordered by increasing complexity of evaluation and the thresholds adjusted so that the early (simpler) tests have few or no false negatives

Any window that is rejected by early tests can be discarded quickly without computing the other features

This is referred to as a **cascade** architecture

Cascading Classifiers

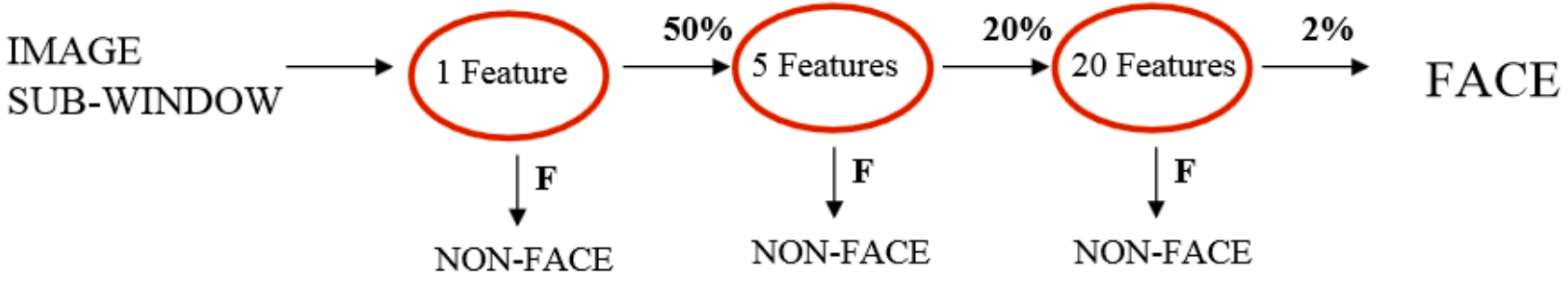


Figure credit: P. Viola

A **classifier** in the cascade is not necessarily restricted to a single feature

Example: Face Detection Summary

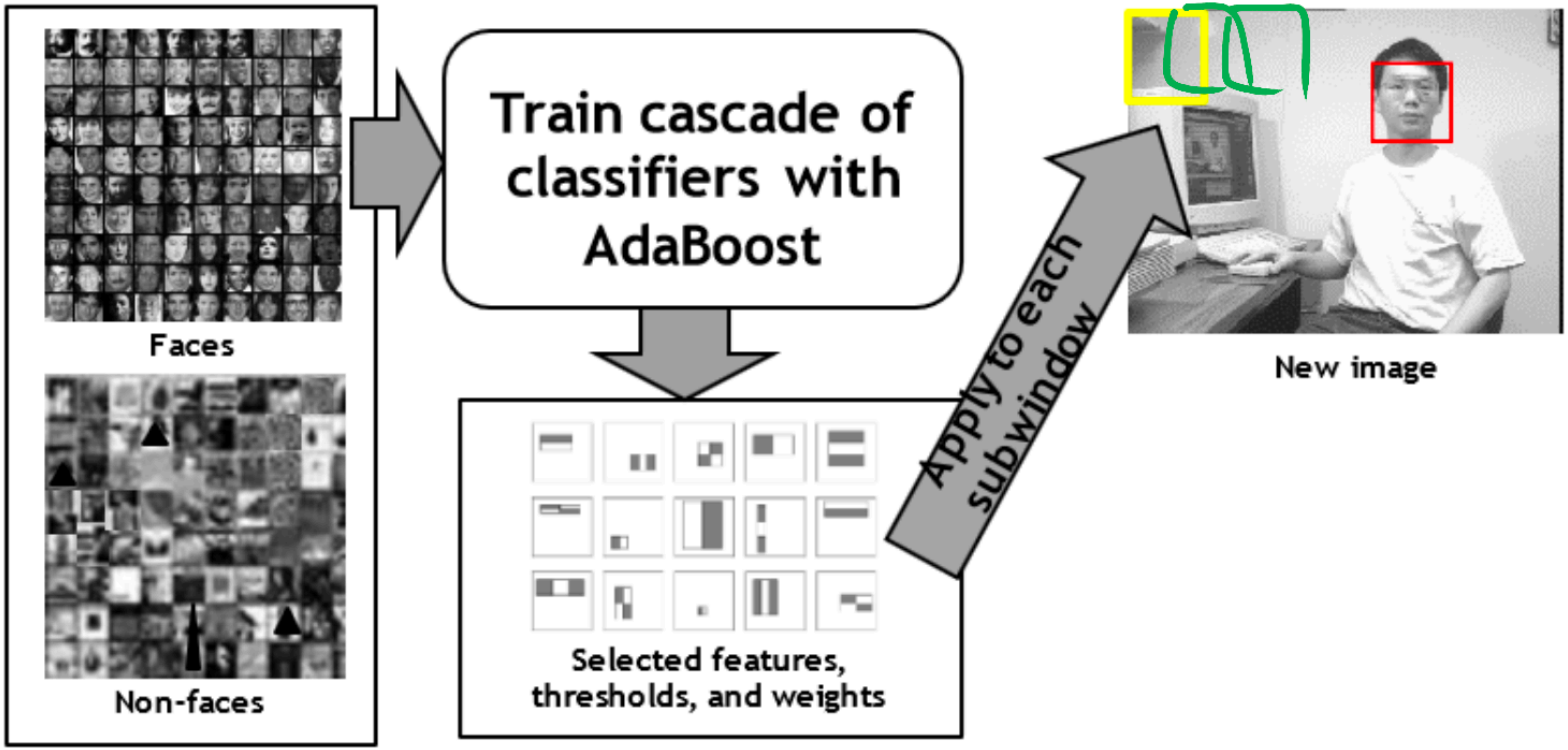


Figure credit: K. Grauman

Hard Negative Mining

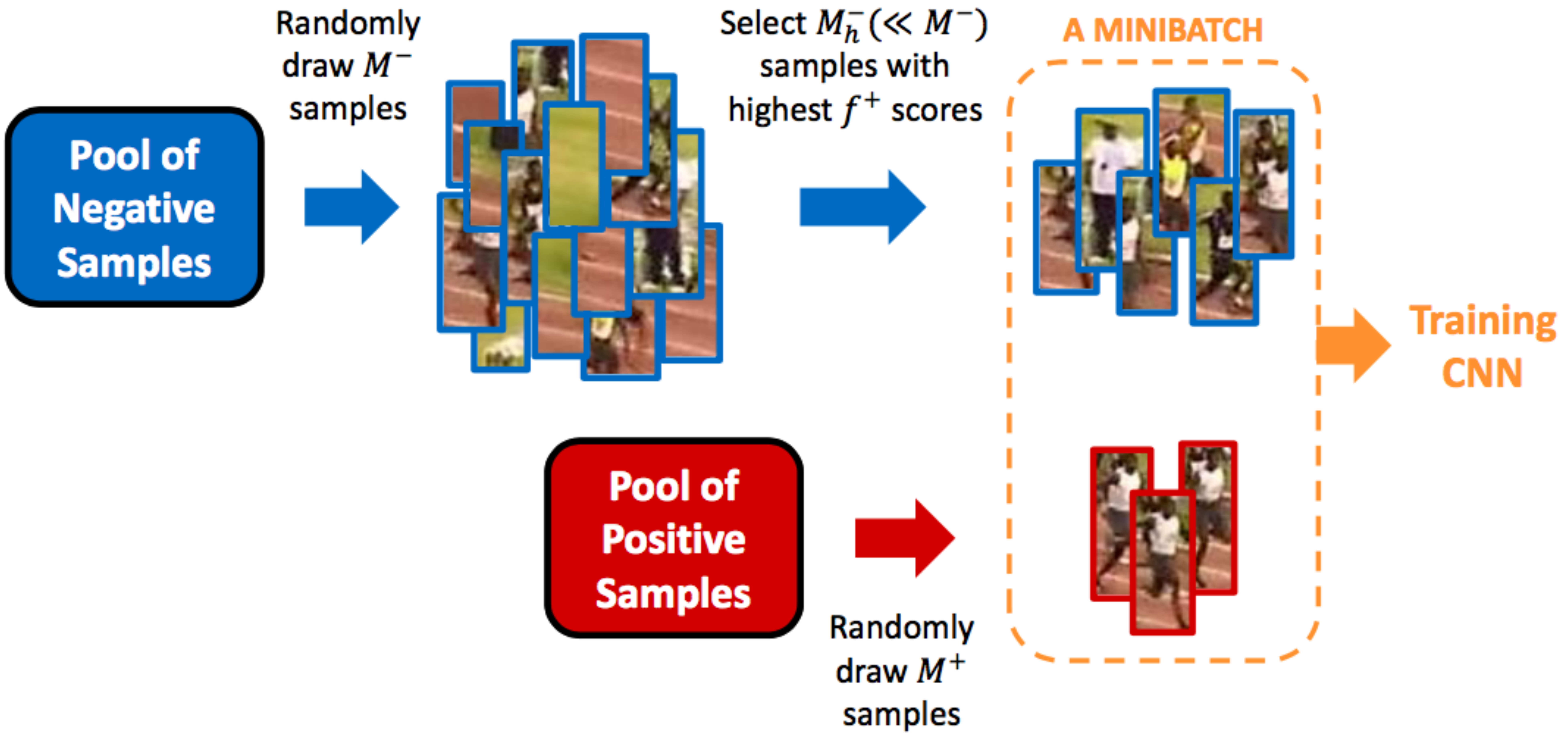
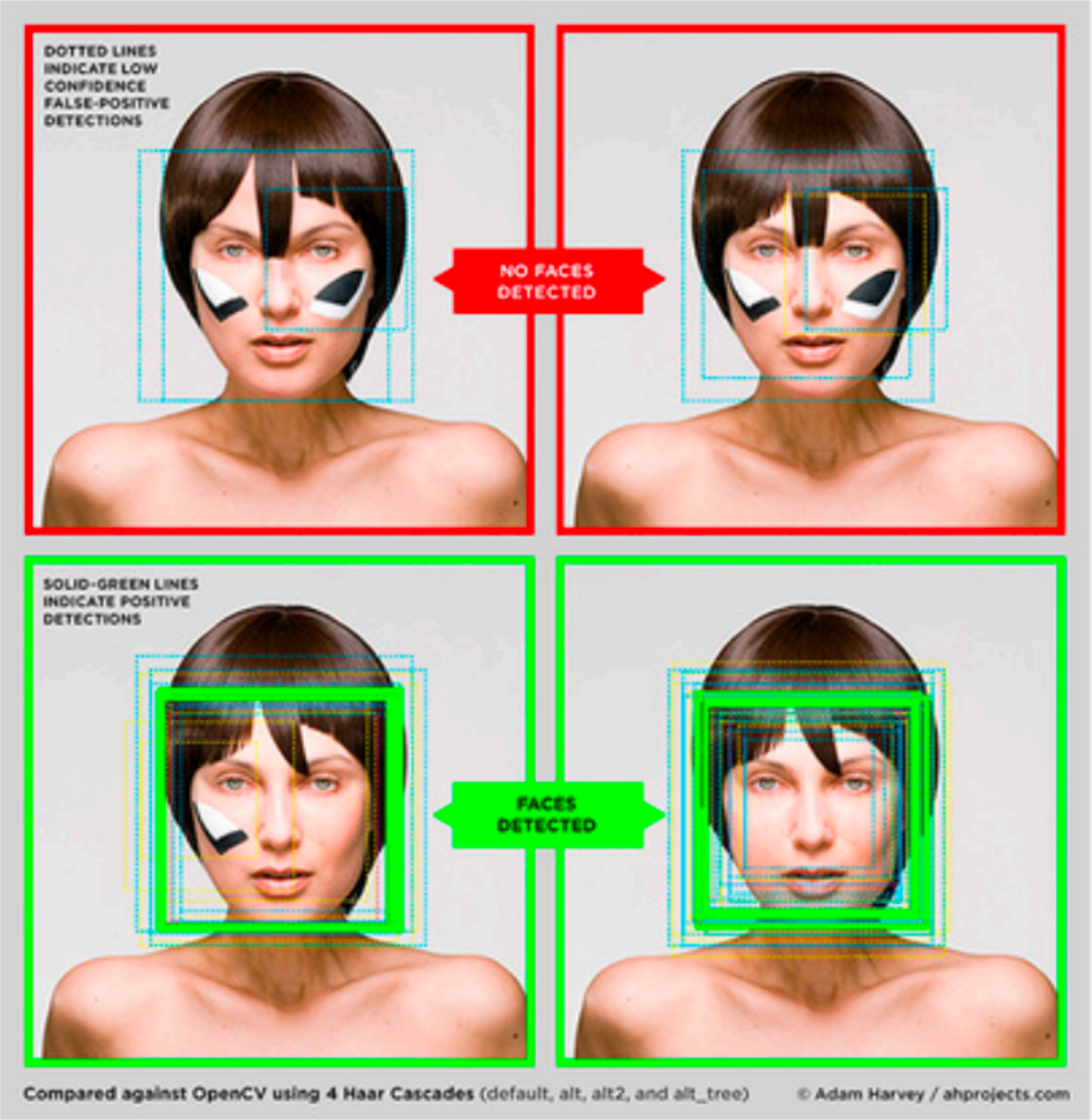


Image From: Jamie Kang

Example: Face Detection

Just for fun:



"CV Dazzle, a project focused on finding fashionable ways to thwart facial-recognition technology"

Figure source: Wired, 2015