

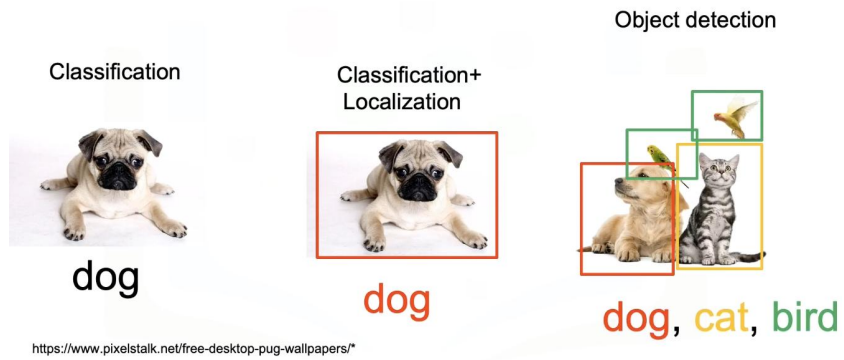
Week 5 - Object Detection

- 5.1 Video: Object Detection
- 5.2 Video: Object Detection with Haar Cascade Classifier
- 5.3 **Lab**: Car Detection with Haar Classifiers
- 5.4 Reading: Object Detection with Deep Learning
- 5.5 **Lab**: Object Detection with Faster R- CNN
- 5.6 **Lab**: Object Detection using Pre-trained Models CV Studio

5.1 Object Detection

Outline

- Sliding Windows
- Bounding Box
- Bounding Box Pipeline
- Score



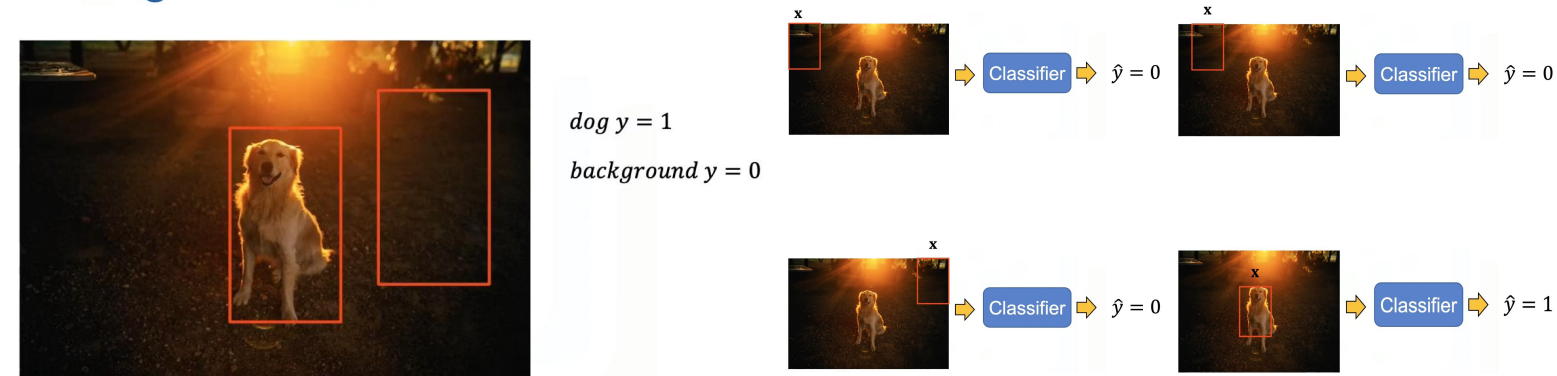
In this video, you will learn about Object Detection, In particular: Sliding Windows, Bounding Box, Bounding Box Pipeline, Score.

Image Classification predicts the class of an object in an image.

Classification and Object Localization locate the presence of an object and indicate the location with a bounding box.

Object detection locates multiple objects with a bounding box and their classes

Sliding Windows



In object detection problems, we generally have to find all the possible objects in the image, to achieve this, we use an algorithm known as **Sliding window detection**. If we want to detect a dog we consider a fixed window size, if chosen properly, the dog will occupy most of the window, this is essentially a sub image that we would like to classify as a dog, the other sub images would be classified as background, each image that does not contain the dog would be considered a background class.

The sliding window algorithm is a more systematic approach.

We start in one region in the image, classify that Sub-image. We then shift the window and classify the next sub-image. We repeat the process. When we get to the horizontal border, we move a few pixels down in the Vertical direction and repeat the process. When the object occupies most of the window, it will be classified as a dog.

Problems of Sliding Windows

- Overlapping Boxes



- Object sizes



Image by: Erikiam, istockphoto.com

• Object shape

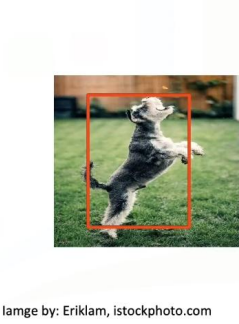
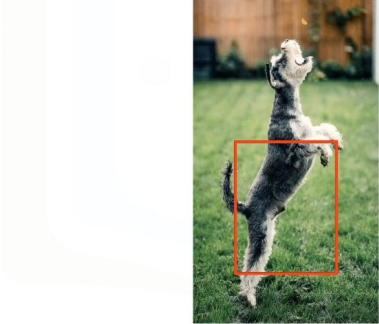
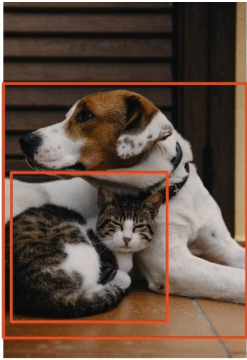


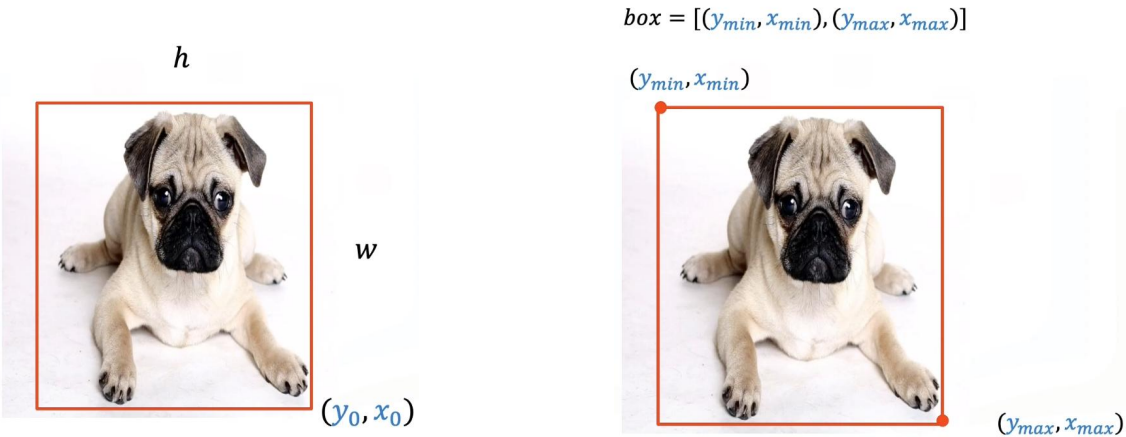
Image by: Eriklam, istockphoto.com

• Overlapping objects



- But in addition to the problems with classification there are additional problems specific to object detection:
- Object detectors often output many overlapping detections.
 - We also have the issue of object sizes, where the same object can come in different sizes, (click 1) one way to solve this is to reshape the image.
 - The same object can have different shapes, again reshaping or resizing the image is one solution.
 - We also have the problem of Overlapping objects in pictures, so as such this may pose issues to the sliding windows.

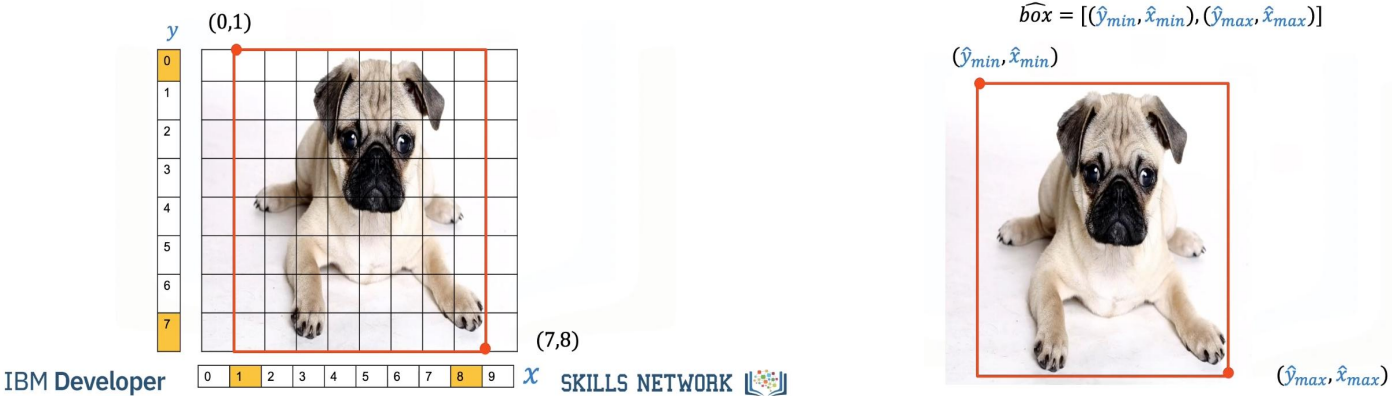
Bounding Box



Bounding Boxes is another method for object Detection, it can be used independently, with sliding Windows or with other more advanced methods.

The bounding box is a rectangular box that can be determined, with the lower-right corner of the rectangle with coordinates y0 and x0 and the width and height. The y and x are not the same as the classification labels y and the image x so we will color them blue.

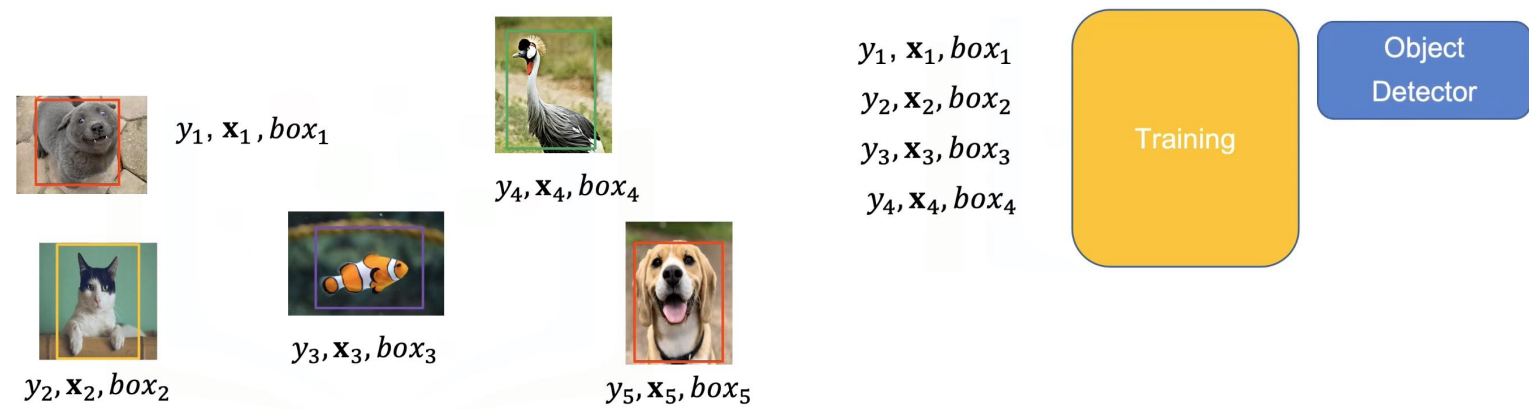
It can also be determined by the coordinates in the upper-left corner y-min and x-min and the lower-right corner the x-max and y-max. **These are not the labels and the image they are just to illustrate the coordinates of the Bounding box we will call box.**



Lets do an example. The coordinates in the upper-left corner is given by 0,1 and the lower-right corner is 7,8.

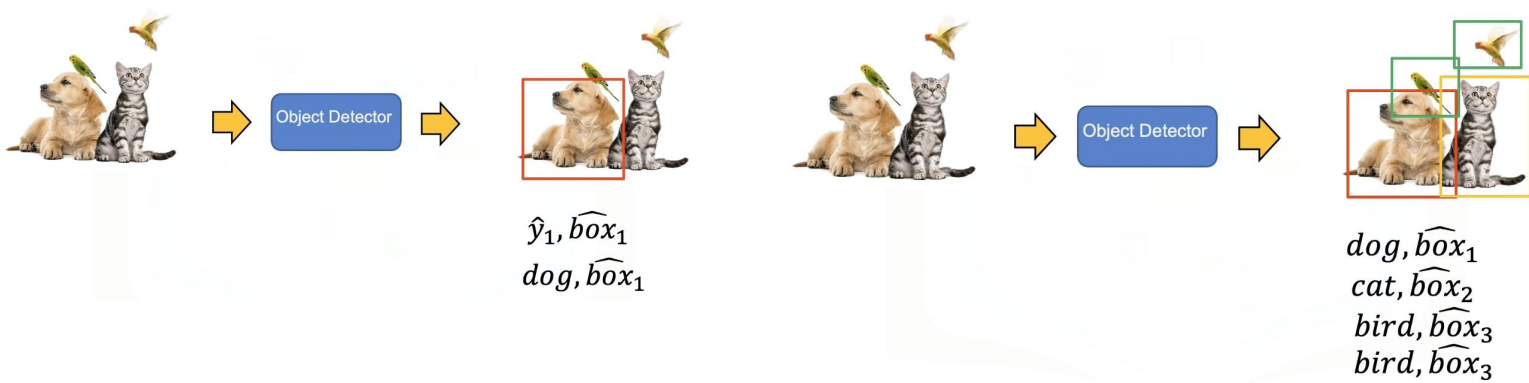
The goal of object detection is to predict these points, so we add a “hat” to indicate it’s a prediction

Bounding Box Pipeline




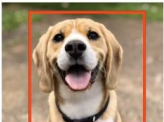
The Bounding Box Pipeline.
Like classification, we have the class y and x . We also have the bounding box, just like classification we have a dataset of Classes and their Bounding boxes.

Similar to classification, we use the dataset to train the model, we include the box coordinates, the result is an object detector with updated learning parameters we will discuss the specific models and training later.

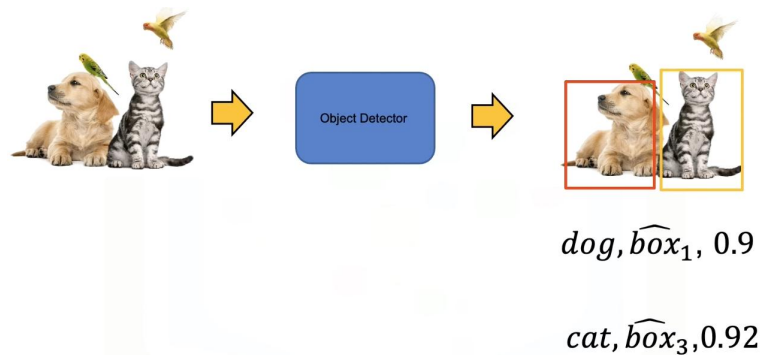
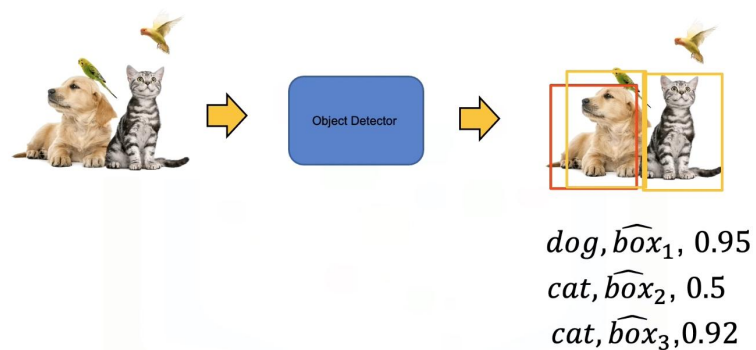
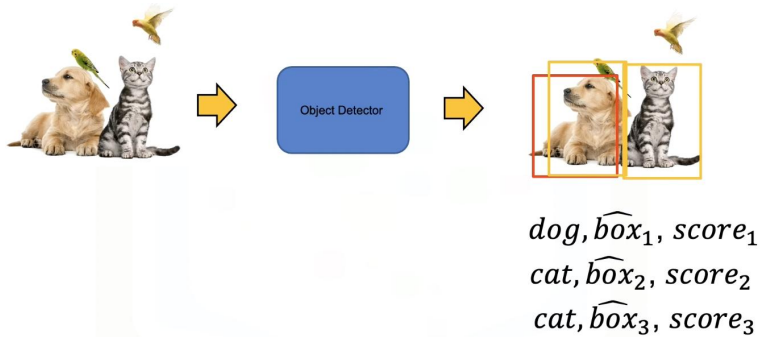


we input the image with the objects we would like to detect. We have the predicted class and the box coordinates, in this case a dog.
We also have the predicted class cat and the box coordinates. The predicted class bird and the box coordinates and another class predicted as bird and the box coordinates.

Score

Score	0.5	0.99
\hat{y}	dog	dog
		

Many object detection algorithms provide a score letting you know how confident the model prediction is. Each column In the table has an image and it's Prediction. The first row is the score ranging from 0 to 1, the second row is the class. And the third row is. the image and its bounding box.
For the first row we see the prediction is dog, but the image does not look like a dog, as a result the score is low 0.5. This means the model is not confident about its prediction. For the second row we see the prediction Is dog and the image looks like a dog, as a result the score is 0.99 in this case, the model is confident about its prediction.



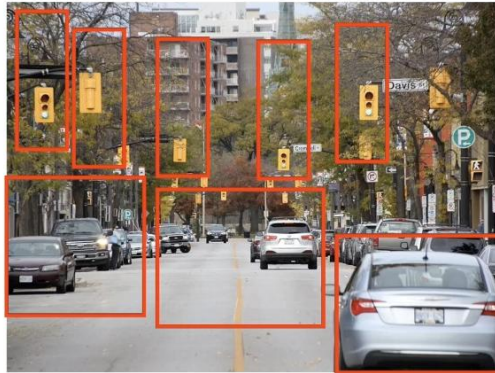
For each detection a score is provided, we can adjust so we only accept detections above a specific score, here we have detected one dog and two cats. It looks like we detected both a cat and a dog in the dogs location.

Examining the score we see that one of the cat prediction's has a low score of 0.5.

If we only accept scores above 0.9 we correctly detect the cat and dog. Usually models will only output objects over a specific threshold. Check out the labs for more examples.

Haar – Cascade Classifiers

- Haar cascade was proposed by P. Viola and M. Jones in 2001
- It is a machine learning method
- Trained on both positive and negative images



In this video we will provide an overview of Object Detection using the Haar feature-based Cascade Classifiers.

We are going to use Haar Feature-based Cascade Classifiers to detect Cars, traffic lights, pedestrian stop signs etc in this image. The method was proposed by P. Viola and M. Jones in 2001.

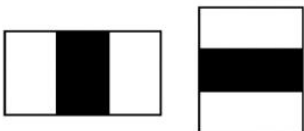
It is a machine learning method where a cascade function is trained on a large number of **positive images which means that it includes the object we are trying to detect** and **negative images i.e the background**.

Haar feature-based Cascade Classifier

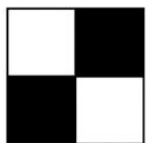
- Based on the Haar wavelet sequence
- Haar like features:



Edge features



Line features



Four-rectangle features



Paul Viola and Michael Jones used the idea of the Haar wavelets in the Haar Feature Classifier.

After millions of training images are fed to the system, the classifier begins by extracting features from each image.

Haar wavelets are convolution kernels used to extract features. Haar wavelets extract information about: Edges, Lines, Diagonal edges. This example we overlay the Haar wavelets over the car

Haar feature-based Cascade Classifier

- The Integral image concept

1	2	2	4	1
3	4	1	5	2
2	3	3	2	4
4	1	5	4	6
6	3	2	1	3



0	0	0	0	0	0
0	1	3	5	9	10
0	4	10	13	22	25
0	6		21	32	39
0	10	20	31	46	59
0	16	29	42	58	74

1	2	2	4	1
3	4	1	5	2
2	3	3	2	4
4	1	5	4	6
6	3	2	1	3



0	0	0	0	0	0
0	1	3	5	9	10
0	4	10	13	22	25
0	6	15	21	32	39
0	10	20	31	46	59
0	16	29	42	58	74

Integral image is a concept that uses the cumulative sum of pixels above and to the left of the current pixel cell

One - The **Integral image concept** is each pixel represents the **cumulative sum** of **the corresponding input pixels above and to the left of that pixel**.

The concept takes in the pixels of an input image as such as this. To get the integral sum for **the highlighted pixel**, we will **add everything to the left and to the top**. Since there is nothing to the left and top, We will get a sum of 1 by just adding itself. To get the value of the 2nd pixel highlighted, we will add 1 which is to the left and itself 2 and get 3. Lets try this once more, to get the integral sum of the highlighted pixel, we will get the sum of 2, 4, 1, 3, 2 and itself, 3. We will get 15.

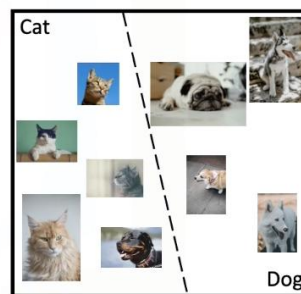
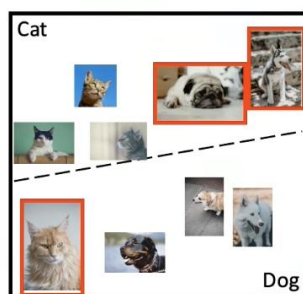
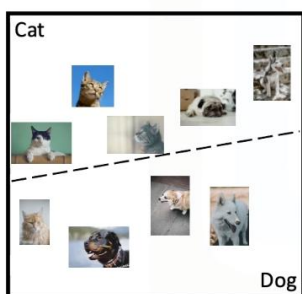
The Viola-Jones paper used a 24 by 24 base window size as an example and that would result in more than 180,000 features calculated in the integral image.

Haar feature-based Cascade Classifier

An AdaBoost classifier is used to reduce the number of features:

- A weak classifier is made on top of the training data based on the weighted samples
- It selects only those features that help to improve the classifier accuracy
- AdaBoost cuts down the number of features significantly

An AdaBoost classifier is used to reduce the number of features:



Two - The algorithm selects a few important features from a large set to give highly efficient classifiers by employing the use of an **AdaBoost**. The idea is to set weights to both classifiers and samples in a way that forces classifiers to concentrate on observations that are difficult to correctly classify. Therefore, it selects only those features that help to improve the classifier accuracy by constructing a strong classifier which is a linear combination of weak classifiers.

In the case of the 24 by 24 window example used by Viola-Jones, over 180000 features were generated. Using the AdaBoost cuts it down to about 6000 features.

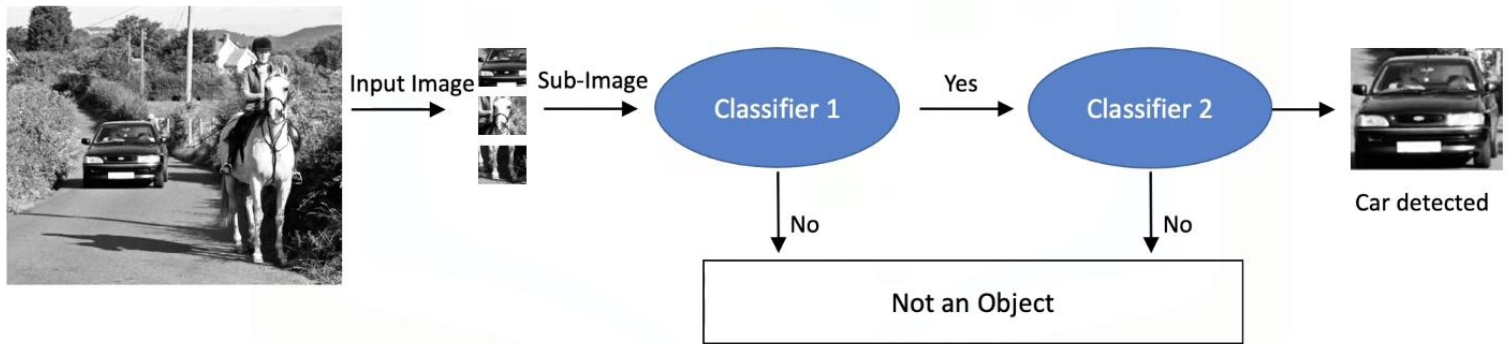
The algorithm selects a few important features from a large set to give highly efficient classifiers by employing the use of an AdaBoost. Let us illustrate with cats and dogs.

Each weak classifier splits the examples with at least 50% accuracy, the misclassified examples are then emphasized on the next round. The idea is to set weights to both classifiers and samples in a way that forces classifiers to concentrate on observations that have been misclassified. The process is repeated until it has minimized the number of errors and constructs a strong classifier. A strong classifier is a linear combination of weak classifiers. In the case of the 24 by 24 window example used by Viola-Jones, over 180,000 features were generated. Using the AdaBoost cuts it down to about 6000 features.

总结:

Haar feature-based Cascade Classifier

A cascade of Classifiers



Three - Cascades of classifiers are then used.

This classifier groups sub-images from the input image in stages and disregards any region that doesn't match the object it is trying to detect. To detect the car in this image, the classifier groups the features into multiple sub-images and the classifier at each stage determines whether the sub-image is the object we are trying to detect.

In the case that it is not, the sub-window is discarded along with the features in that window. If the sub-window moves past the classifier, it continues to the next stage where the second stage of features is applied until it is sure that it is a car.

5.3 Lab: Car Detection with Haar Classifiers

另见下载下来的网页 5.3_Car Detection with Haar Classifiers

5.4 Reading: Object Detection with Deep Learning

另见下载下来的网页 5.4_Reading_Object Detection with Deep Learning

5.5 Lab: Object Detection with Faster R- CNN

详见 5.5_Object Detection with Faster R- CNN.ipynb

5.6 Lab: Object Detection using Pre-trained Models CV Studio

另见下载下来的网页 5.6_Object Detection using Pre-trained Models CV Studio

总结：