2.1.1: An introduction to object detection

Topic Progress: (https://gurus.pyimagesearch.com/topic/an-introduction-to-object-detection/)

(https://gurus.pyimagesearch.com/topic/template-matching/)

← Back to Lesson (https://gurus.pyimagesearch.com/lessons/what-are-object-detectors/)

In our **Image Classification module (https://gurus.pyimagesearch.com/topic/what-is-image-classification/)** we have started to explore computer vision and machine learning techniques to *classify the contents of an image*.

Using these computer vision and machine learning algorithms, we can train a classifier that can be used to categorize and label the contents of an image. For example, let's pretend that we trained a classifier to determine if an image contains a stop sign or not. When presented with the following image:

(https://gurus.pyimagesearch.com/wp-content/uploads/2015/03/what_is_object_detection_stop_sign.jpg)

FIGURE 1: AN EXAMPLE IMAGE CONTAINING A STOP SIGN. USING SIMPLE IMAGE CLASSIFICATION WE CAN REPORT THAT THE IMAGE DOES INDEED CONTAIN A STOP SIGN — BUT *WHERE* IN THE IMAGE IS THE STOP SIGN?

We can leverage our trained classifier to (theoretically) correctly categorize this image as *stop sign*.

However, just because we can report an image as containing a stop sign *does not* mean we can report *exactly where in the image* the stop sign is.

As humans, it's trivial for us to (1) identify the stop sign in the image, and (2) note that the location of the stop sign is halfway down the image at the far right. But for a computer, this localization task is not so easy. Is the stop sign at the center of the image? At the top of the image? In the left corner? Right corner? Even if we were given the quadrant of the image the stop sign was contained in, how might we obtain a bounding box of the stop sign, such as coordinates *(112, 50)* with a width and height of 85 pixels?

Simple image classification algorithms can only give us a *global labeling and categorization* of an image. They *cannot* provide local labelings of the image and tell us where the stop sign is, where the railroad is, etc.:

(https://gurus.pyimagesearch.com/wp-content/uploads/2015/03/what_is_object_detection_localization.jpg)

FIGURE 2: BASIC IMAGE CLASSIFICATION ALGORITHMS CAN ONLY REPORT ON WHETHER OR NOT AN IMAGE CONTAINS A SPECIFIC OBJECT. THEY CANNOT DETAIL EXACTLY *WHERE* IN THE IMAGE THE OBJECT IS.

Feedback

For a more granular classification of an image, such as identifying each of the "objects" in an image, we need to perform ***object detection***.

Objectives:

In this lesson we will learn about the basics of object detection; specifically, the difference between global image classification and local object detection.

Object Detectors

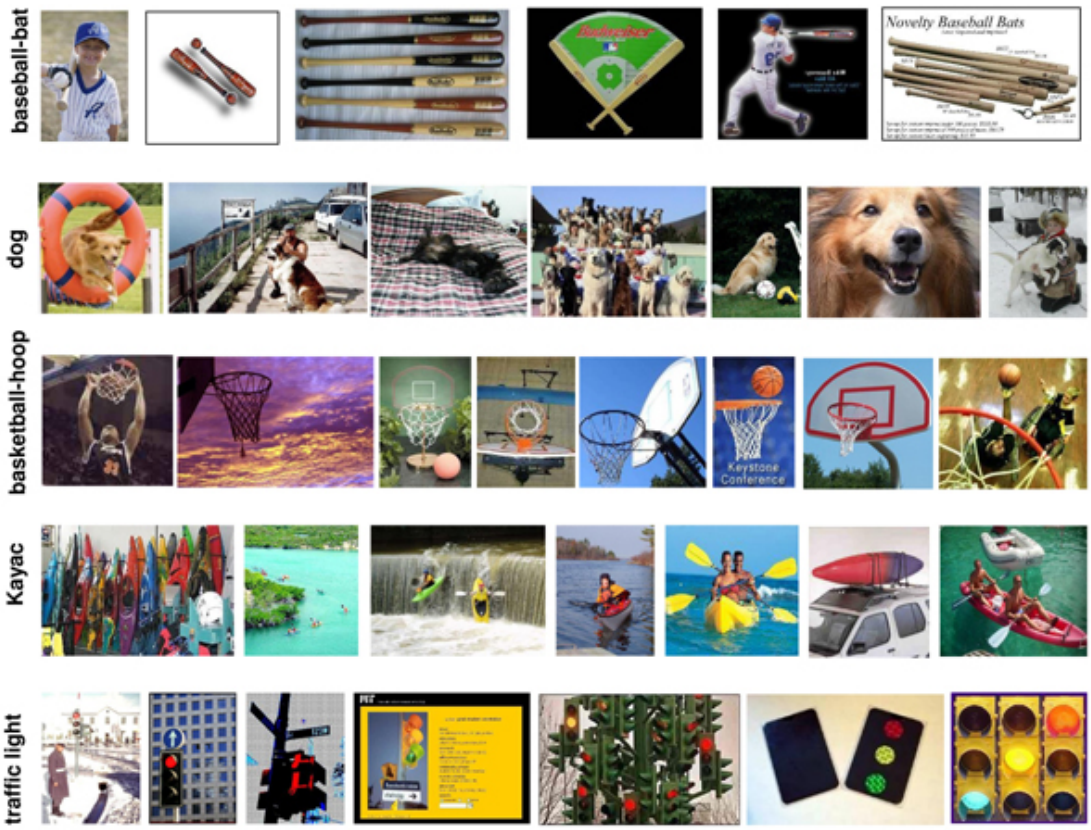
When building an object detector, the term "object" is used fast and loose — and if the term sounds a bit abstract, it's because it is.

An object can be a chair, a person, or even a glass of beer:

(https://gurus.pyimagesearch.com/wp-content/uploads/2015/03/what_is_object_detection_example_objects.jpg)

FIGURE 3: OBJECTS IN IMAGES ARE "ABSTRACT" BY DEFINITION. OBJECTS CAN BE A PERSON, A CHAIR, A OR EVEN A GLASS OF BEER.

In general, any physical entity with a *semi-rigid structure* (meaning the object is not overly deformable and can dramatically alter its shape) can be considered an "object". Here are a few examples of objects from CALTECH-101 (http://www.vision.caltech.edu/Image_Datasets/Caltech101/), a popular 101-category object detection benchmark dataset:



(https://gurus.pyimagesearch.com/wp-content/uploads/2015/03/what_is_object_detection_caltech101.jpg)

FIGURE 4: EXAMPLES IMAGES FROM THE CALTECH-101 DATASET (SOURCE (HTTP://WWW.ROBOTS.OX.AC.UK:5000/~VGG/RESEARCH/CALTECH/INDEX.HTML)).

Feedback

In the figure above you can see examples of dogs, traffic lights, kayaks, baseball bats, etc. — all of these entities are considered "objects" since they are represented as a semi-rigid structure with noticeable (and repeatable) patterns that our computer vision and machine learning algorithms can detect.

What makes object recognition so hard?

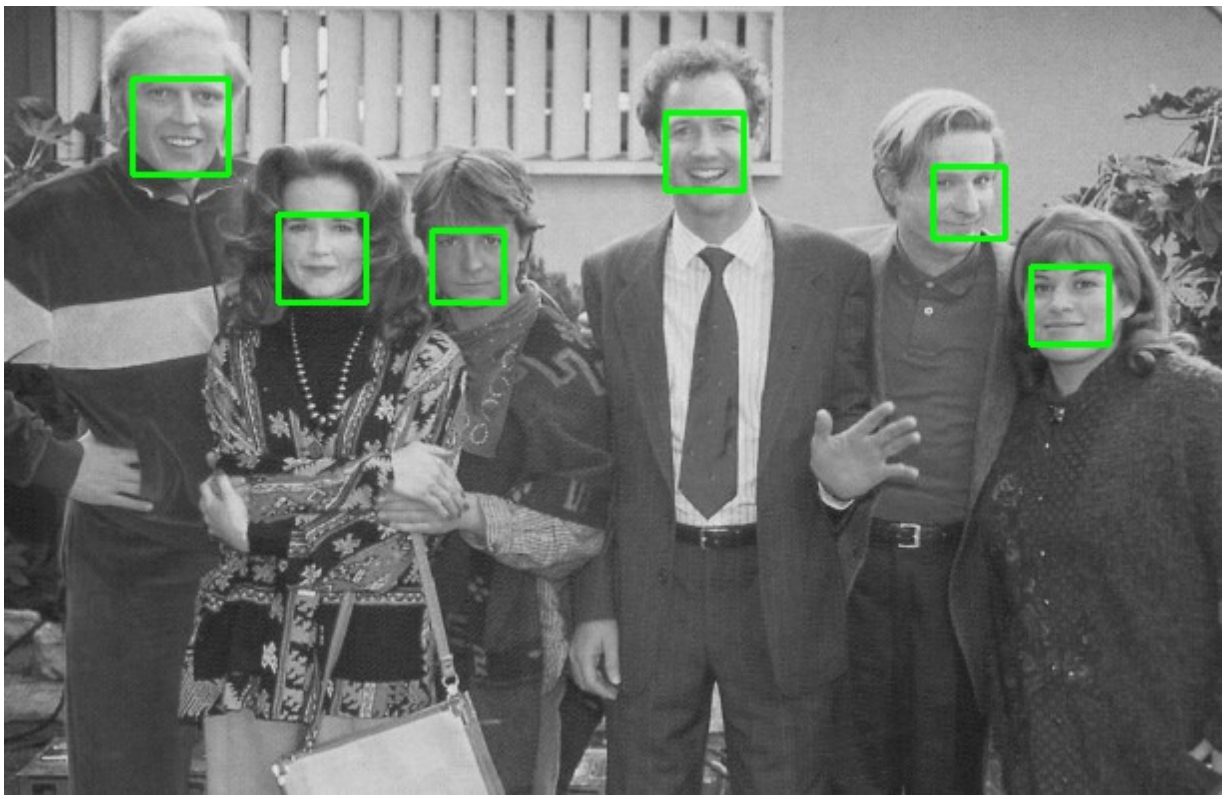
Object recognition is hard for the same reason **image classification is hard**. Objects in the real-world can exhibit substantial variations in *viewpoint*, *scale*, *deformation*, *occlusion*, *illumination*, *background clutter*, and *intra-class variation* (just like **Figure 4** above demonstrates) — all of which we discuss in more detail in the **What is image classification? (https://gurus.pyimagesearch.com/topic/what-is-image-classification/)** lesson.

A good object detector should be robust to changes in these properties and still be able to detect the presence of the object, even under less-than-ideal circumstances.

Where is object detection used in the real-world?

Object detection is used in your every day life, whether you realize it or not.

For example, ***detecting the presence of faces in images*** is a form of object detection:



(https://gurus.pyimagesearch.com/wp-content/uploads/2015/03/what_is_object_detection_faces.jpg)

FIGURE 5: FACE DETECTION IS AN EXAMPLE OF OBJECT DETECTION.

A face represents a rigid, predicable object structure and pattern: two eyes, two ears on either side, a nose below the eyes, lips below the nose, and a chin below the lips. Since nearly all faces share these traits, we thus have a common structure and pattern that we can detect in images.

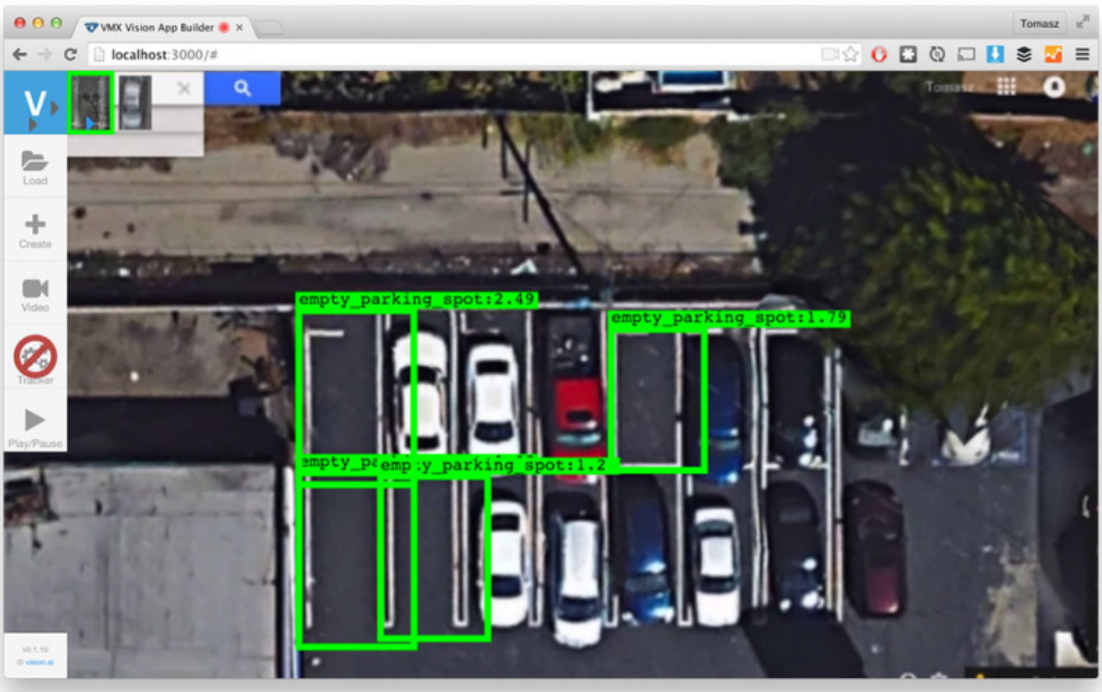
Face detection is used all the time, but you're probably most familiar with the implementation in your digital camera or smartphone — face detection can be used to perform auto-focusing to ensure the face(s) are clear in the shot.

We can also use object detection in ***security systems*** where we track people in video feeds and monitor their activity:

(https://gurus.pyimagesearch.com/wp-content/uploads/2015/03/what_is_object_detection_security_feed.jpg)

FIGURE 6: SECURITY SYSTEMS USED TO TRACK SUSPECTS IN LIVE FEED VIDEOS IS ALSO A FORM OF OBJECT DETECTION — THE SUSPECT MUST BE *DETECTED* BEFORE THEY CAN BE *TRACKED*.

Another great real-world implementation of object detection is ***automated vehicle parking garages*** where computer vision techniques can be used to detect if a parking spot is open or not:



(https://gurus.pyimagesearch.com/wp-content/uploads/2015/03/what_is_object_detection_parking_spots.jpg)

FIGURE 7: IT MAY SEEM STRANGE THINKING OF AN EMPTY PARKING SPOT AS AN "OBJECT", BUT WE CAN ACTUALLY MODEL PARKING SPOTS AS OBJECTS USING COMPUTER VISION AND MACHINE LEARNING (SOURCE (HTTPS://TWITTER.COM/QUANTOMBONE/STATUS/614501075480133633)).

Finally, object detection is used all the time in ***visual search***. In the example image below, we have applied object detection to identify each piece of clothing in the image (i.e. shirt, jeans, etc.):

(https://gurus.pyimagesearch.com/wp-content/uploads/2015/03/what_is_object_detection_fashion.jpg)

FIGURE 8: OBJECT DETECTION CAN BE LEVERAGED TO DETECT VARIOUS ITEMS OF CLOTHING — AND THEN USE THESE DETECTIONS TO BUILD A FASHION SEARCH ENGINE USING ONLY VISUAL CUES.

Given each of the pieces of clothing, we can then apply visual search algorithms to find similar clothing items online.

Summary

As we already know from our module on **Image Classification** (https://gurus.pyimagesearch.com/topic/what-is-image-classification/), we can use computer vision and machine learning algorithms to build a classifier to label/categorize an image. However, just because we can obtain this categorization *does not* mean we can pinpoint the location of a specific object in an image.

In order to determine the exact location of an object in an image, we need to extend our image classification knowledge and construct an *object detector*. As the name suggests, an object detector is used to scan an image and look for the presence of a given object (i.e. a stop sign, motorcycle, dog, etc.)

As we'll find out in the rest of this module, object detectors tend to be substantially more challenging to build than simple image classifiers; however, the results we obtain from our object detectors are often times much more useful.

In our next lesson we'll start off with the most basic form of object detection — *template matching*.

	Quizzes	Status
1	An Introduction to Object Detection Quiz (https://gurus.pyimagesearch.com/quizzes/an-introduction-to-object-detection-quiz/)	

Next Topic → (https://gurus.pyimagesearch.com/topic/template-matching/)

Course Progress

Ready to continue the course?

Click the button below to **continue your journey to computer vision guru**.

I'm ready, let's go! (/pyimagesearch-gurus-course/)

Resources & Links

- PyImageSearch Gurus Community (https://community.pyimagesearch.com/)
- PyImageSearch Virtual Machine (https://gurus.pyimagesearch.com/pyimagesearch-virtual-machine/)
- Setting up your own Python + OpenCV environment (https://gurus.pyimagesearch.com/setting-up-your-python-opencv-development-environment/)
- Course Syllabus & Content Release Schedule (https://gurus.pyimagesearch.com/course-syllabus-content-release-schedule/)
- Member Perks & Discounts (https://gurus.pyimagesearch.com/pyimagesearch-gurus-discounts-perks/)

- Your Achievements (https://gurus.pyimagesearch.com/achievements/)
- Official OpenCV documentation (http://docs.opencv.org/index.html)

Your Account

- Account Info (https://gurus.pyimagesearch.com/account/)
- Support (https://gurus.pyimagesearch.com/contact/)
- Logout (https://gurus.pyimagesearch.com/wp-login.php?action=logout&redirect_to=https%3A%2F%2Fgurus.pyimagesearch.com%2F&_wpnonce=5736b21cae)

Search

Feedback

/