MathWorks®
Help Center

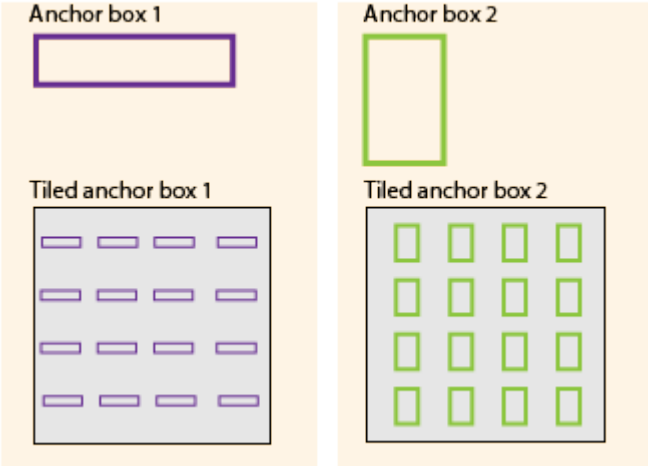
Anchor Boxes for Object Detection

Object detection using deep learning neural networks can provide a fast and accurate means to predict the location and size of an object in an image. Ideally, the network returns valid objects in a timely manner, regardless of the scale of the objects. The use of anchor boxes improves the speed and efficiency for the detection portion of a deep learning neural network framework.

What Is an Anchor Box?

Anchor boxes are a set of predefined bounding boxes of a certain height and width. These boxes are defined to capture the scale and aspect ratio of specific object classes you want to detect and are typically chosen based on object sizes in your training datasets. During detection, the predefined anchor boxes are tiled across the image. The network predicts the probability and other attributes, such as background, intersection over union (IoU) and offsets for every tiled anchor box. The predictions are used to refine each individual anchor box. You can define several anchor boxes, each for a different object size. Anchor boxes are fixed initial boundary box guesses.

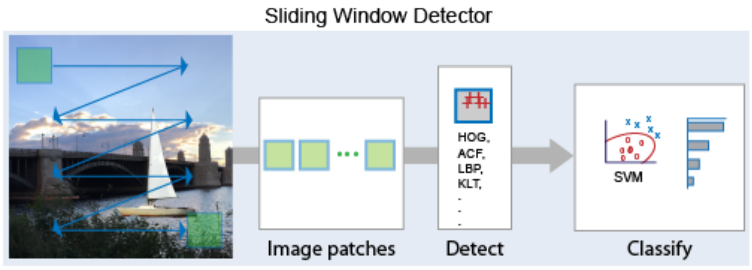
The network does not directly predict bounding boxes, but rather predicts the probabilities and refinements that correspond to the tiled anchor boxes. The network returns a unique set of predictions for every anchor box defined. The final feature map represents object detections for each class. The use of anchor boxes enables a network to detect multiple objects, objects of different scales, and overlapping objects.



Advantage of Using Anchor Boxes

When using anchor boxes, you can evaluate all object predictions at once. Anchor boxes eliminate the need to scan an image with a sliding window that computes a separate prediction at every potential position. Examples of detectors that use a sliding window are those

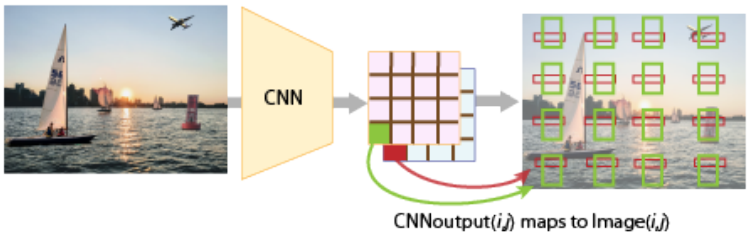
that are based on aggregate channel features (ACF) or histogram of gradients (HOG) features. An object detector that uses anchor boxes can process an entire image at once, making real-time object detection systems possible.



Because a convolutional neural network (CNN) can process an input image in a convolutional manner, a spatial location in the input can be related to a spatial location in the output. This convolutional correspondence means that a CNN can extract image features for an entire image at once. The extracted features can then be associated back to their location in that image. The use of anchor boxes replaces and drastically reduces the cost of the sliding window approach for extracting features from an image. Using anchor boxes, you can design efficient deep learning object detectors to encompass all three stages (detect, feature encode, and classify) of a sliding-window based object detector.

How Do Anchor Boxes Work?

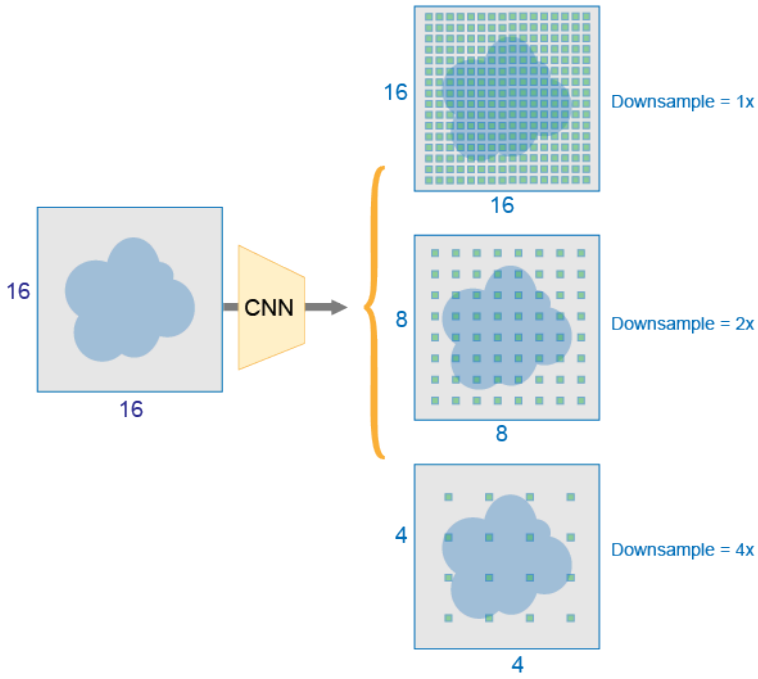
The position of an anchor box is determined by mapping the location of the network output back to the input image. The process is replicated for every network output. The result produces a set of tiled anchor boxes across the entire image. Each anchor box represents a specific prediction of a class. For example, there are two anchor boxes to make two predictions per location in the image below.



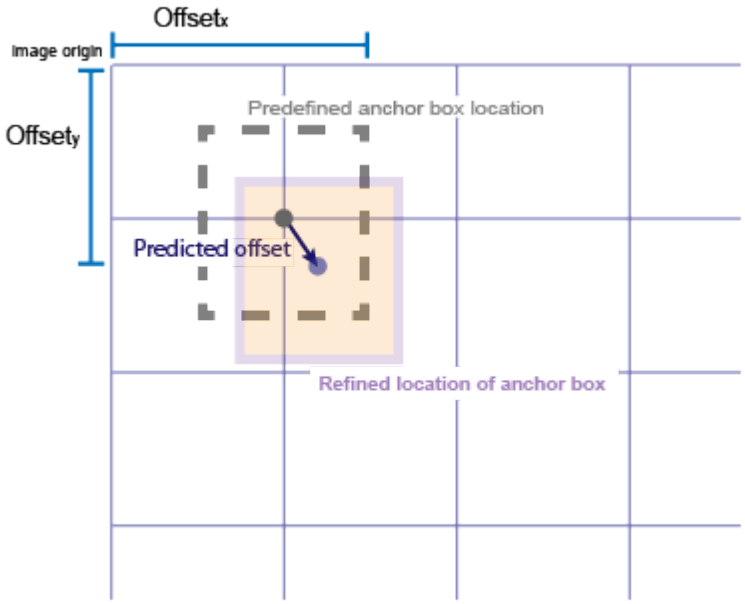
Each anchor box is tiled across the image. The number of network outputs equals the number of tiled anchor boxes. The network produces predictions for all outputs.

Localization Errors and Refinement

The distance, or *stride*, between the tiled anchor boxes is a function of the amount of downsampling present in the CNN. Downsampling factors between 4 and 16 are common. These downsampling factors produce coarsely tiled anchor boxes, which can lead to localization errors.



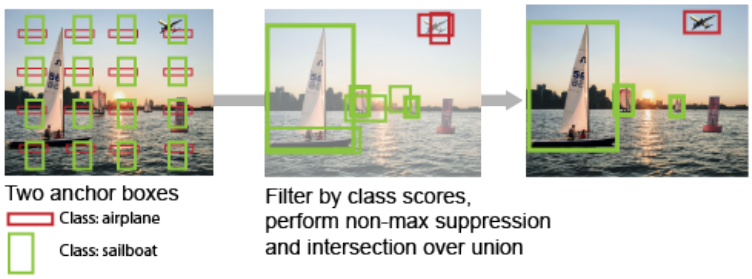
To fix localization errors, deep learning object detectors learn offsets to apply to each tiled anchor box refining the anchor box position and size.



Downsampling can be reduced by removing downsampling layers. To reduce downsampling, decrease the `Stride` property value of the convolution or max pooling layers, such as `convolution2dLayer` (Deep Learning Toolbox) and `maxPooling2dLayer` (Deep Learning Toolbox). You can also choose a feature extraction layer earlier in the network. Feature extraction layers from earlier in the network have a higher spatial resolution but may extract less semantic information compared to layers further down the network.

Generate Object Detections

To generate the final object detections, tiled anchor boxes that belong to the background class are removed, and the remaining ones are filtered by their confidence score. Anchor boxes with the greatest confidence score are selected using nonmaximum suppression (NMS). For more details about NMS, see the `selectStrongestBboxMulticlass` function.



Anchor Box Size

Multiscale processing enables the network to detect objects of varying size. To achieve multiscale detection, you must specify anchor boxes of varying size, such as 64-by-64, 128-by-128, and 256-by-256. Specify sizes that closely represent the scale and aspect ratio of objects in your training data. For an example of estimating sizes, see Estimate Anchor Boxes From Training Data.

Related Examples

- Object Detection Using YOLO v4 Deep Learning
- Estimate Anchor Boxes From Training Data

More About

- Getting Started with YOLO v4
- Deep Learning in MATLAB (Deep Learning Toolbox)
- Pretrained Deep Neural Networks (Deep Learning Toolbox)