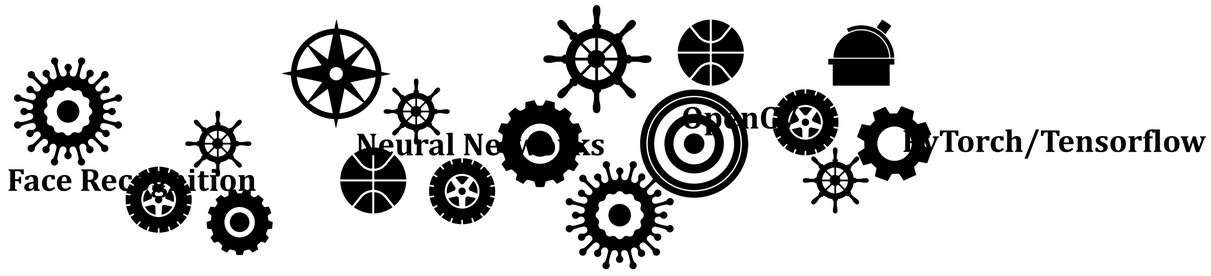
MPSSDEGB
कौशल से रोजगार, समृद्धि अपार

FOUNDATION FOR INNOVATION AND TECHNOLOGY TRANSFER
भारतीय प्रौद्योगिकी संस्थान दिल्ली
Indian Institute of Technology Delhi

Computer Vision

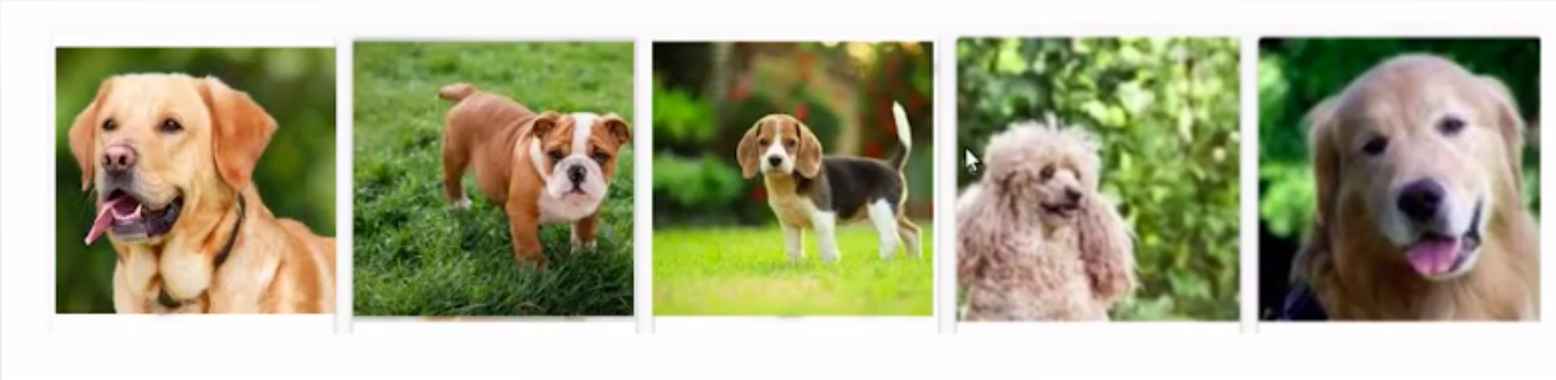


Deep Learning
Segmentation
Object Detection
Image Classification

OUTLINE

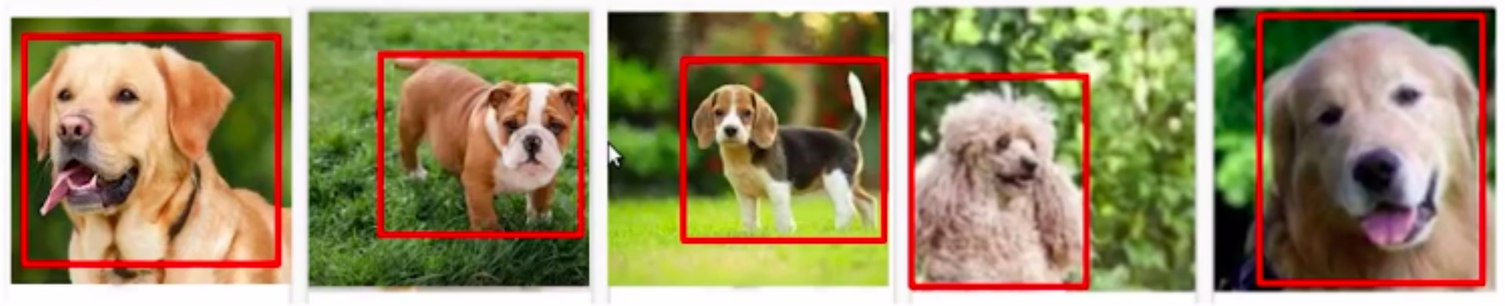
- Introduction to Computer Vision
- Object Detection
- Image Classification Vs Object Detection
- Object Detection Problem
- Basic Concept of Image in Object Detection
- Various Object Detection Methods

Object Detection



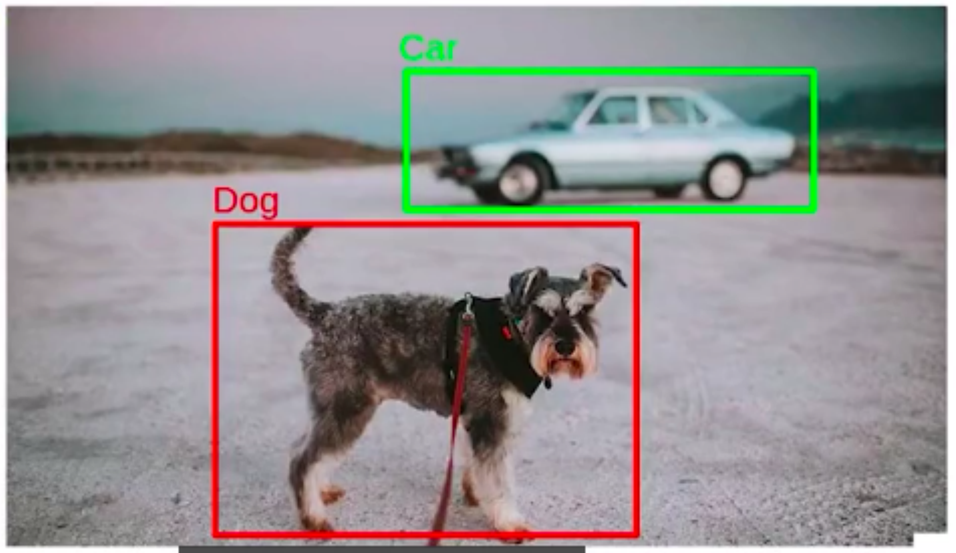
Object detection is a computer vision technique for locating instances of objects in images or videos. Humans can easily detect and identify objects present in an image.

Object Detection - Bounding Box



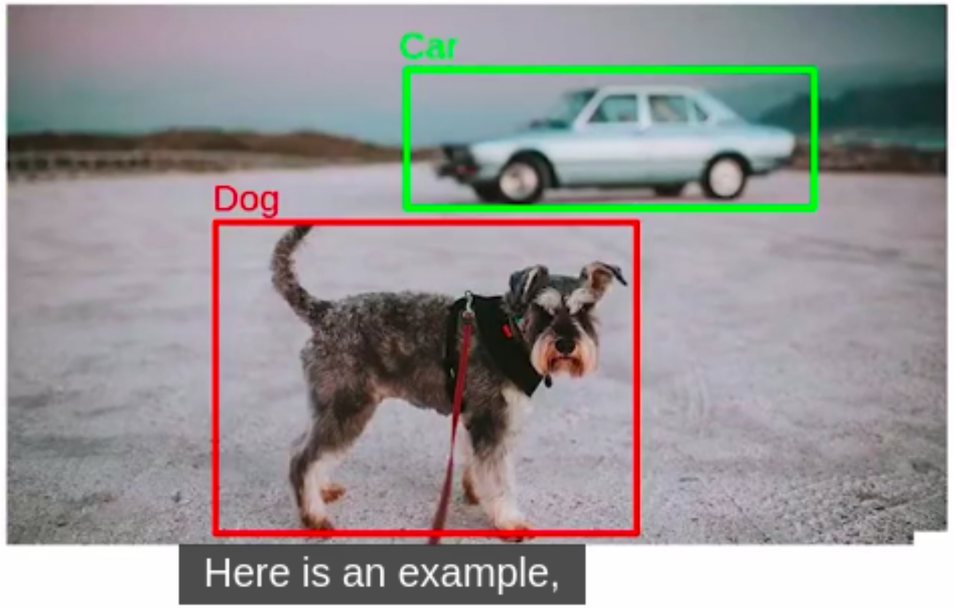
we can create a box around the dog that is present in the image and specify the x and y coordinates of this box.

Object Detection - Bounding Box



Consider that the location of the object in the image can be represented as coordinates of these boxes.
This box around the object in the image is formally known as a bounding box.
It has become an image localization problem where we are given a set of images and we have to identify where is the object present in the image.

Bounding Box-Example



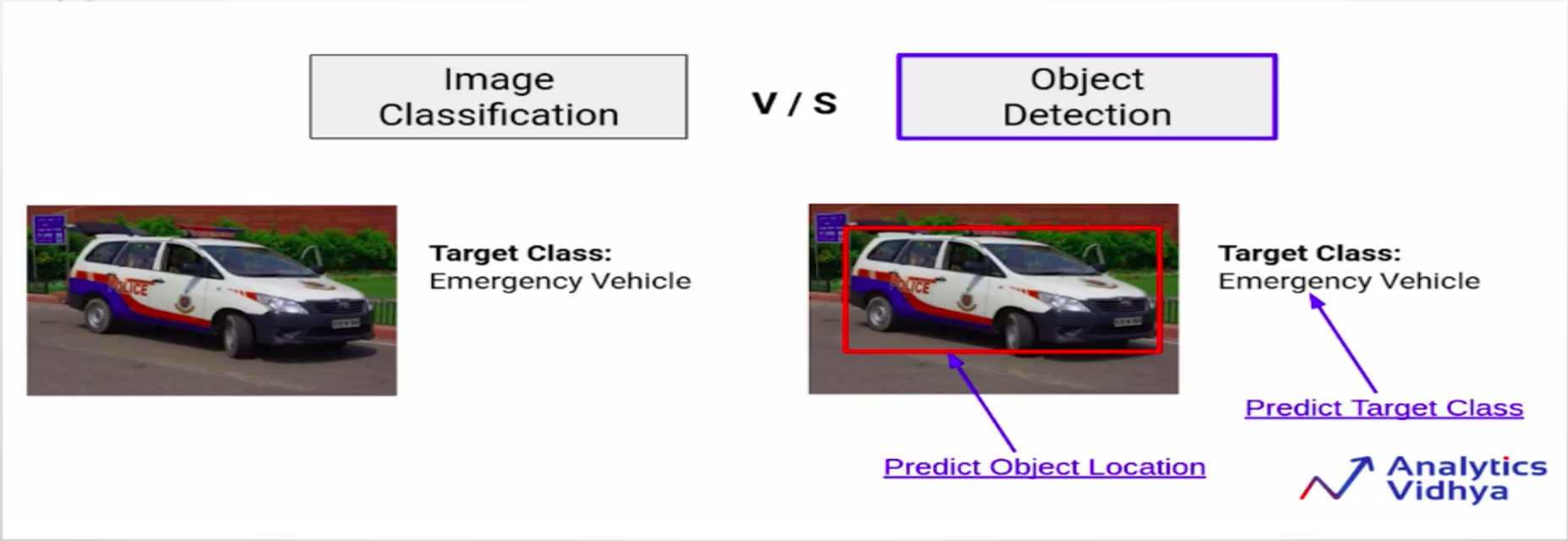
We have to locate the objects in the image but note that all the objects are not dogs.

There is a car and dog in the image

We not only have to locate the objects in the image but also classify the located object as a dog or Car.

This becomes an object detection problem.

Image classification v/s object detection-Example



In the first case, we predict only the target class, and such tasks are known as image classification problems.
In the second case, along with predicting the target class, we also have to find the bounding box which denotes the location of the object.

Image classification v/s object detection

Image Classification	V / S	Object Detection
• Object Classification		• Object Classification • Object Localization

In the case of object detection problems, we have to classify the objects in the image and also locate where these objects are present in the image.

Object detection problem

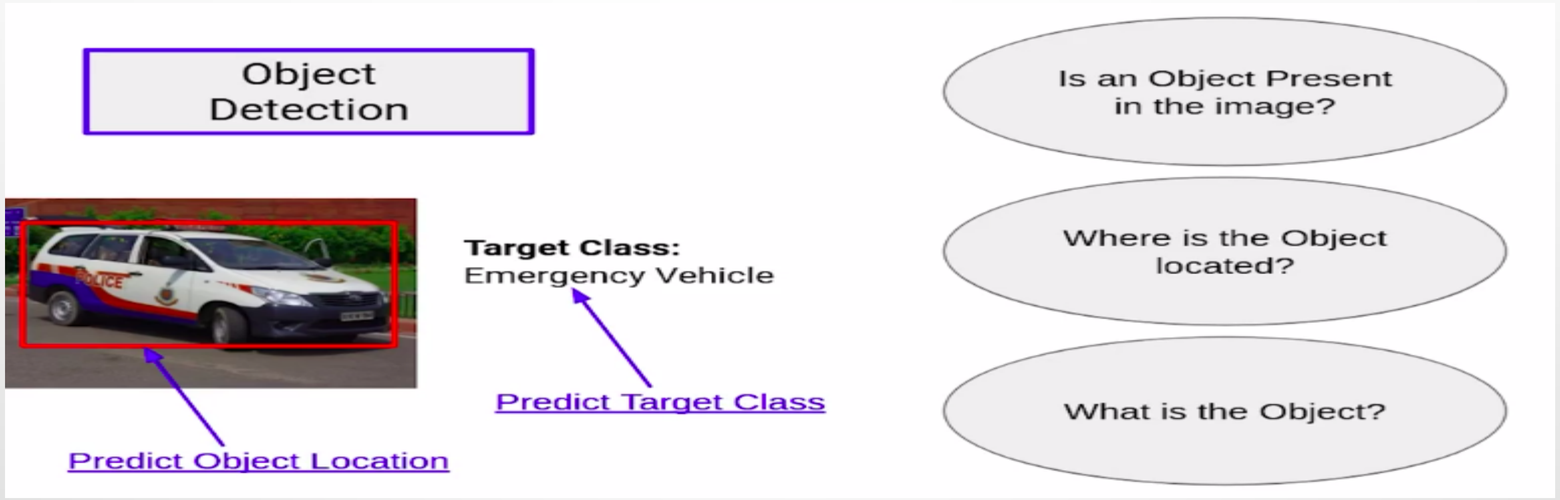
Three tasks for object detection

1. To identify if there is an object present in the image,

2. where is this object located,

3. what is this object?

Object detection problem



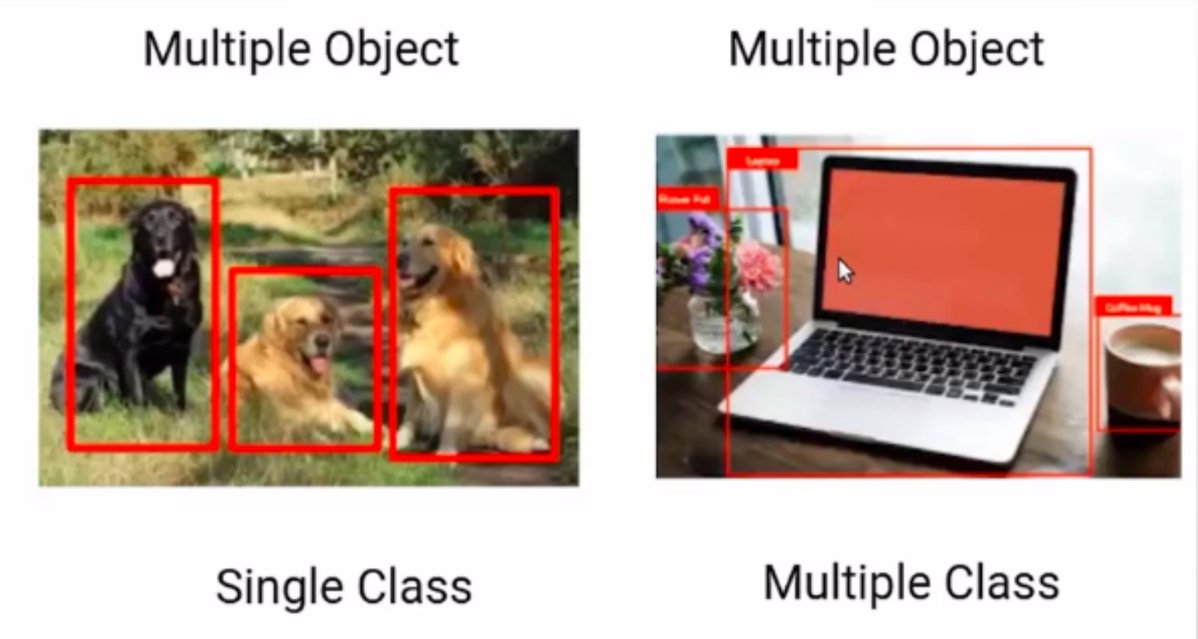
Three tasks for object detection

1. To identify if there is an object present in the image,

2. where is this object located,

3. what is this object?

Multi - Object detection problem



When the dataset has multi object, these objects can be of the same class, or another problem can be that these objects are of different classes.

Training Data For Object Detection

Y = 1

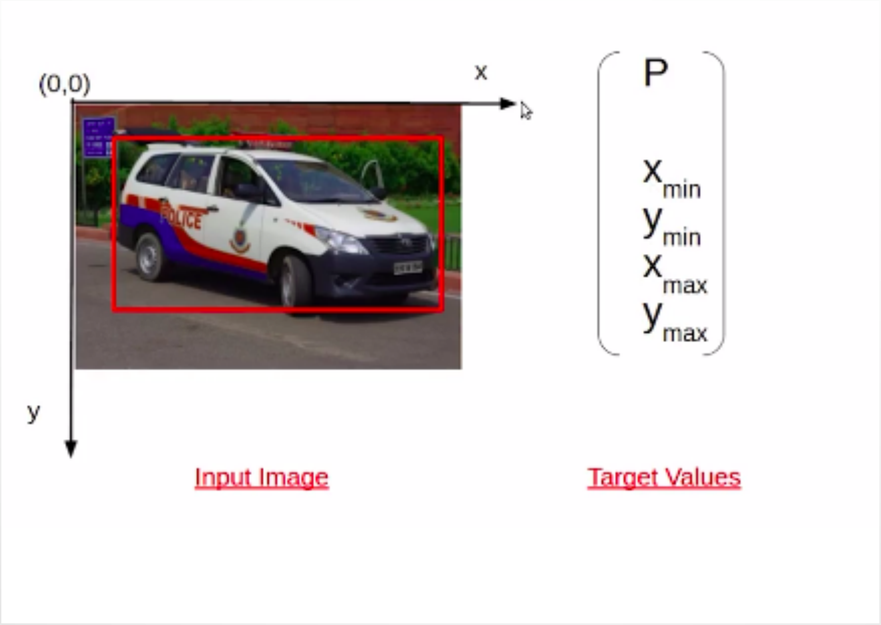
Input Image

Target Class

Data would look like for an object detection task. An example from the classification problem

we have an input image and a target class against each of these input images.

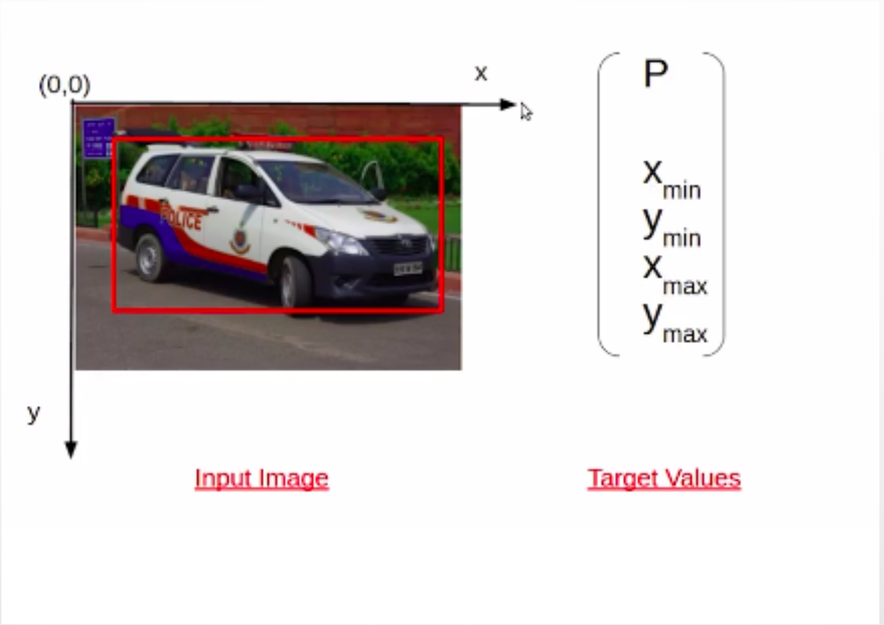
Training Data For Object Detection-Example



Example of detecting the car

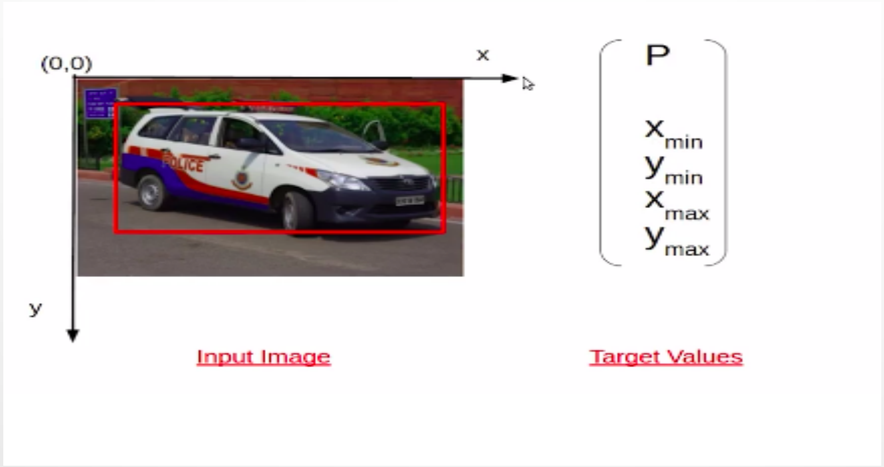
So in that case will not only have an input image but along with a target variable that has the bounding box that denotes the location of the object in the image.

Training Data For Object Detection-Example



The target variable has five values the value p denotes the probability of an object being in the above image whereas the four values Xmin, Ymin, Xmax, and Ymax denote the coordinates of the bounding box.

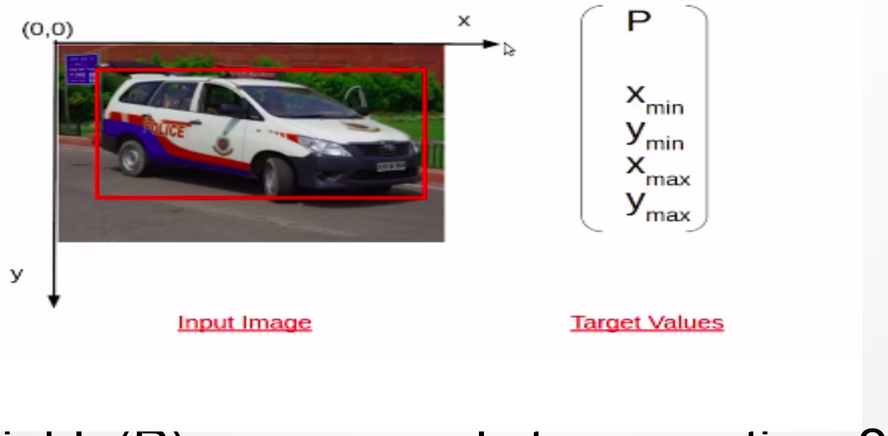
Training Data For Object Detection-Example



Consider the x-axis and y-axis above the image

Xmin and Ymin will be the top left corner of the bounding box, and Xmax and Ymax will be the bottom right corner of the bounding box.

Training Data For Object Detection-Example



Target variable(P) answers only two questions?
1. Is there an object present in the image?

Answer:- If an object is not present then p will be zero and when there is an object present in the image p will be one.

2. if an object is present in the image where is the object located?

Answer:- You can find the object location using the coordinates of the bounding box.

Training Data For Object Detection-Example

Sample Input Images

$$\begin{pmatrix} P \\ x_{min} \\ y_{min} \\ x_{max} \\ y_{max} \\ c_1 \\ c_2 \end{pmatrix}$$

if you have two classes which are an emergency vehicle and a non-emergency vehicle, you'll have two additional values c1 and c2 denoting which class does the object present in the above image belong.

Training Data For Object Detection-Example

Sample Input Images

$$\begin{pmatrix} P \\ x_{min} \\ y_{min} \\ x_{max} \\ y_{max} \\ c_1 \\ c_2 \end{pmatrix}$$

Lets consider this example, we have the probability of an object present in the image as one. We have the given Xmin, Ymin, Xmax, and Ymax as the coordinates of the bounding box. And then we have c1 is equal to 1 since this is an emergency vehicle and c2 would be 0 because of a non-emergency vehicle.

Training Data For Object Detection-Example

Input Image

Target Values	Input Target	Predictions
P	1	0.8
x_{min}	40	70
y_{min}	20	30
x_{max}	210	220
y_{max}	180	190
c_1	1	1
c_2	0	0

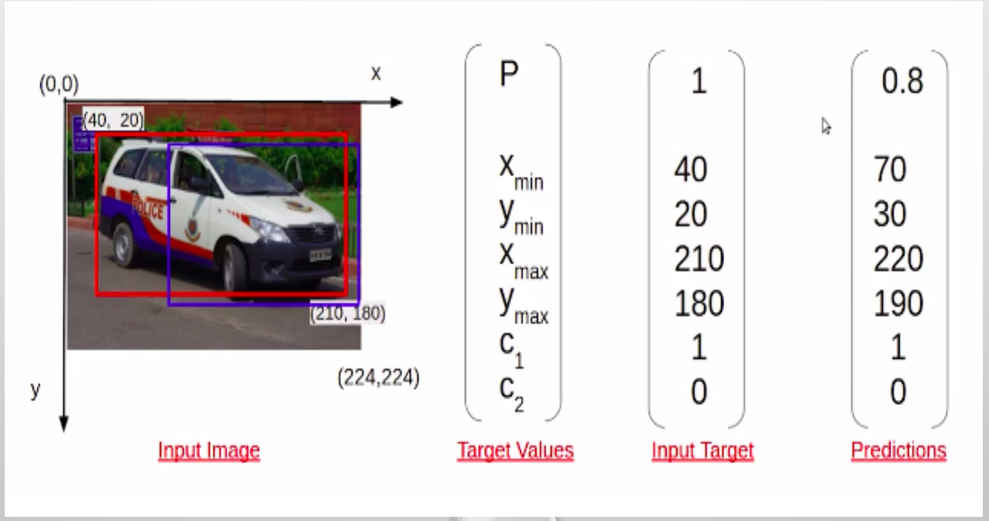
Target Values | Input Target | Predictions

Training Data For Object Detection-Example

we should build a model and get some predictions from the model, this is a possible output that you can get from a model.

The probability that an object is present in this predicted bounding box is given as 0.8.

We have the coordinates of this blue bounding box, which is the predicted bounding box which is (40,20) and (210,180), and then finally the class value of c1 and c2.



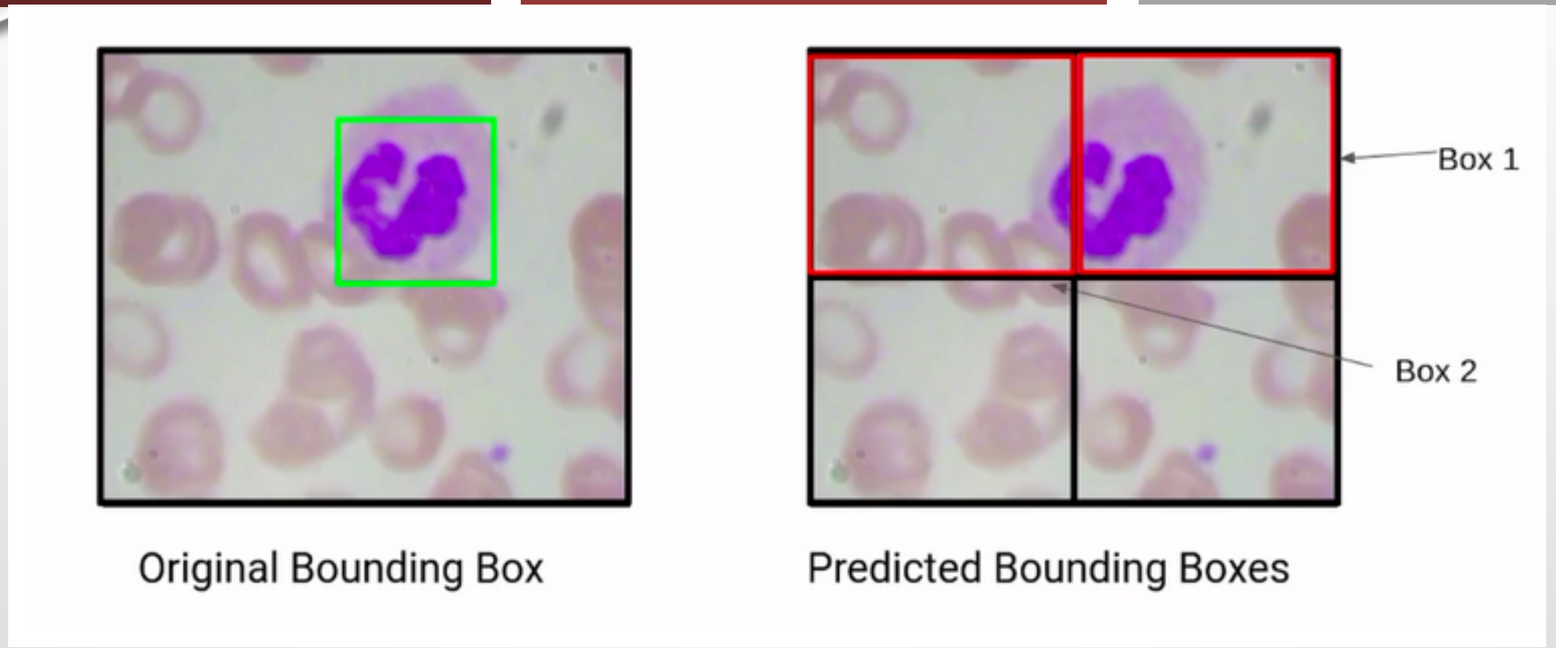
Basic concept of Image in Object Detection

1. How to do Bounding Box Evaluation?

2. How to calculate IoU?

3. Evaluation Metric – mean Average Precision

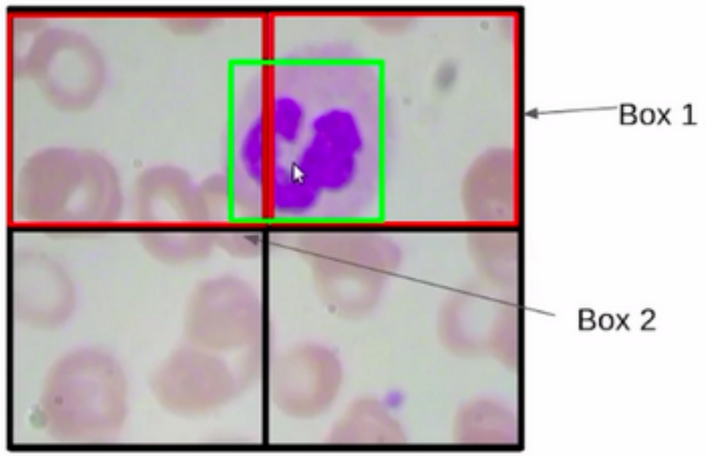
Bounding Box Evaluation – Intersection over Union (IoU)



union(IoU) - is used to determine the target variable for the individual patches that we have created.

Bounding Box Evaluation – Intersection over Union (IoU)

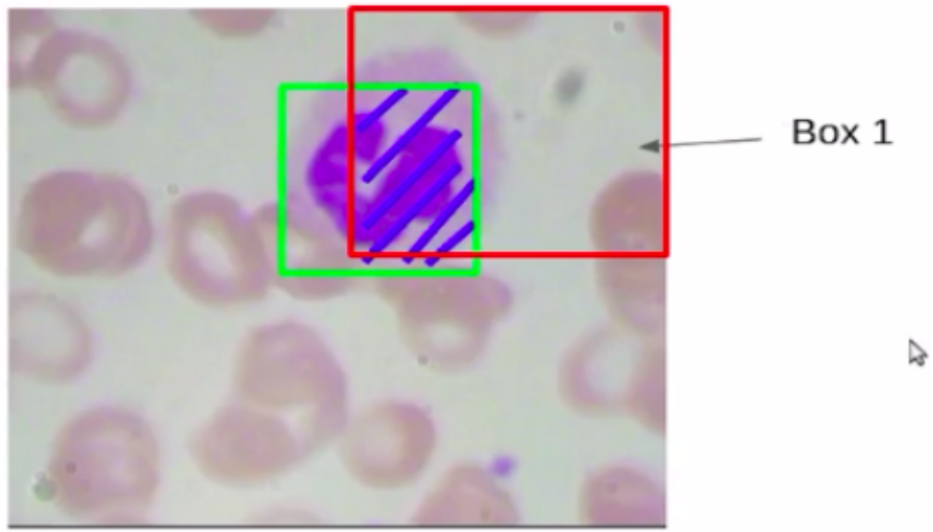
Which Bounding Box is more accurate?



Predicted Bounding Boxes

Consider the following scenario. Here we have two bounding boxes, box1 and box2. Now if I ask you which of these two boxes is more accurate, the obvious answer is box1.

Bounding Box Evaluation – Intersection over Union (IoU)

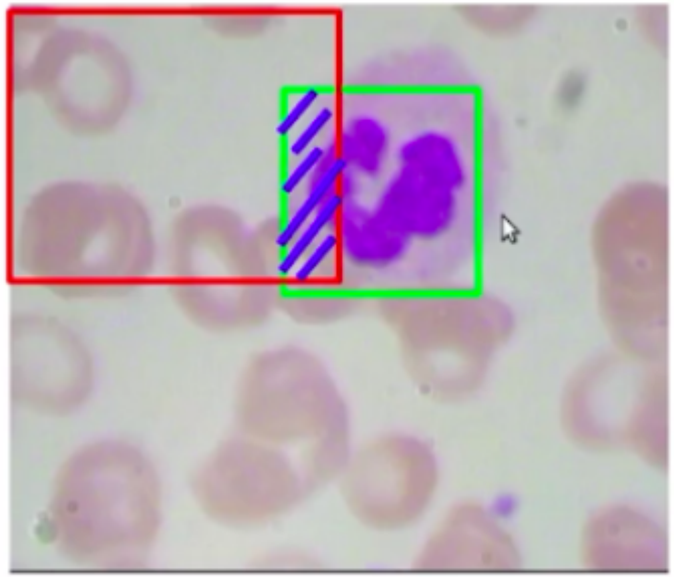


Predicted Bounding Boxes

It has a major region of the WBC and has correctly detected the WBC.

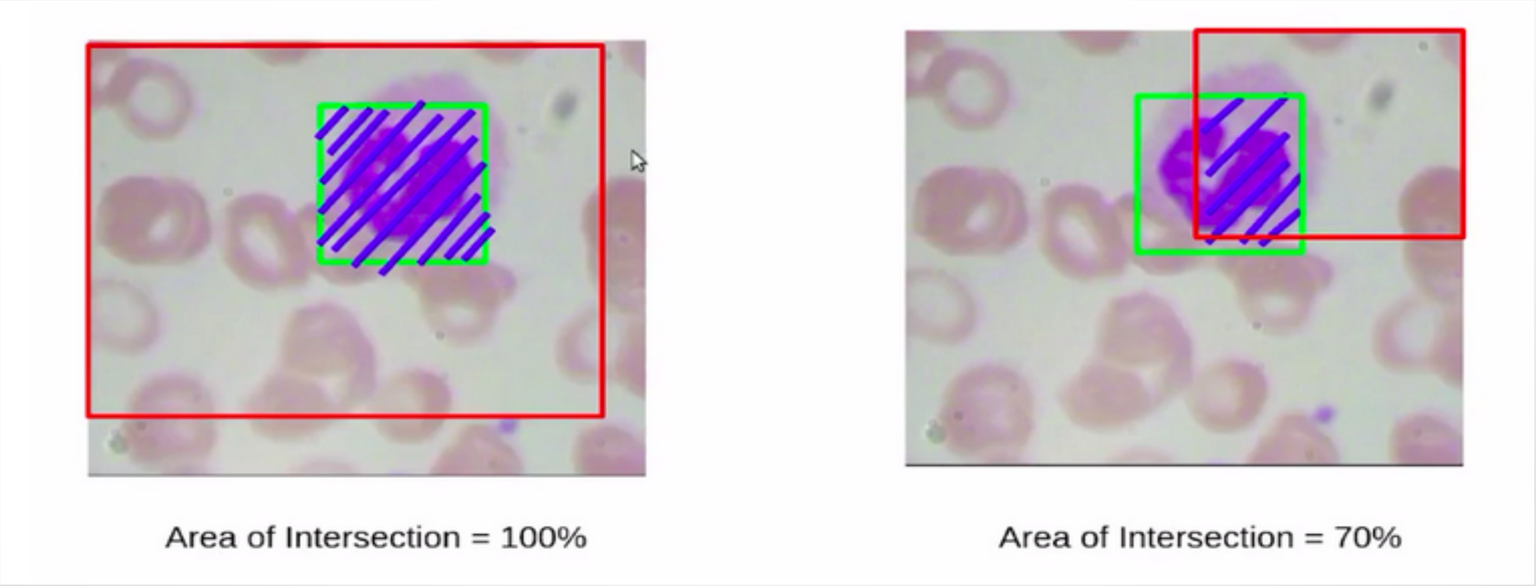
Compare the actual, and the predicted bounding boxes. if we are able to find out the overlap of the actual, and the predicted bounding box, we will be able to make a decision as to which bounding box is a better prediction.

Bounding Box Evaluation – Intersection over Union (IoU)



The bounding box that has a higher overlap with the actual bounding box is a better prediction.
Now, this overlap is called the area of intersection for this first box, which is box1.
We can say that the area of intersection is about 70% of the actual bounding box.
Whereas, if you consider box2, the area of intersection of the second bounding box, and the actual bounding box is about 20 %.

Bounding Box Evaluation – Intersection over Union (IoU)



Scenarios:- 1

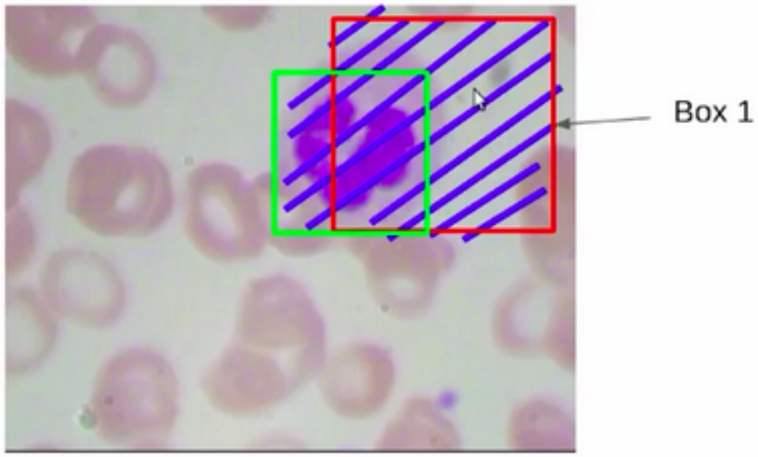
Let’s consider another example suppose we have created multiple bounding boxes or patches of different sizes.

Here, the intersection of the left bounding box is certainly 100% whereas, in the second image, the intersection of this predicted bounding box, or this particular patch is just 70%.
So at this stage, would you say that the bounding box on the left is a better prediction? obviously not. The bounding box on the right is more accurate.

Bounding Box Evaluation – Intersection over Union (IoU)

Which Bounding Box is more accurate?

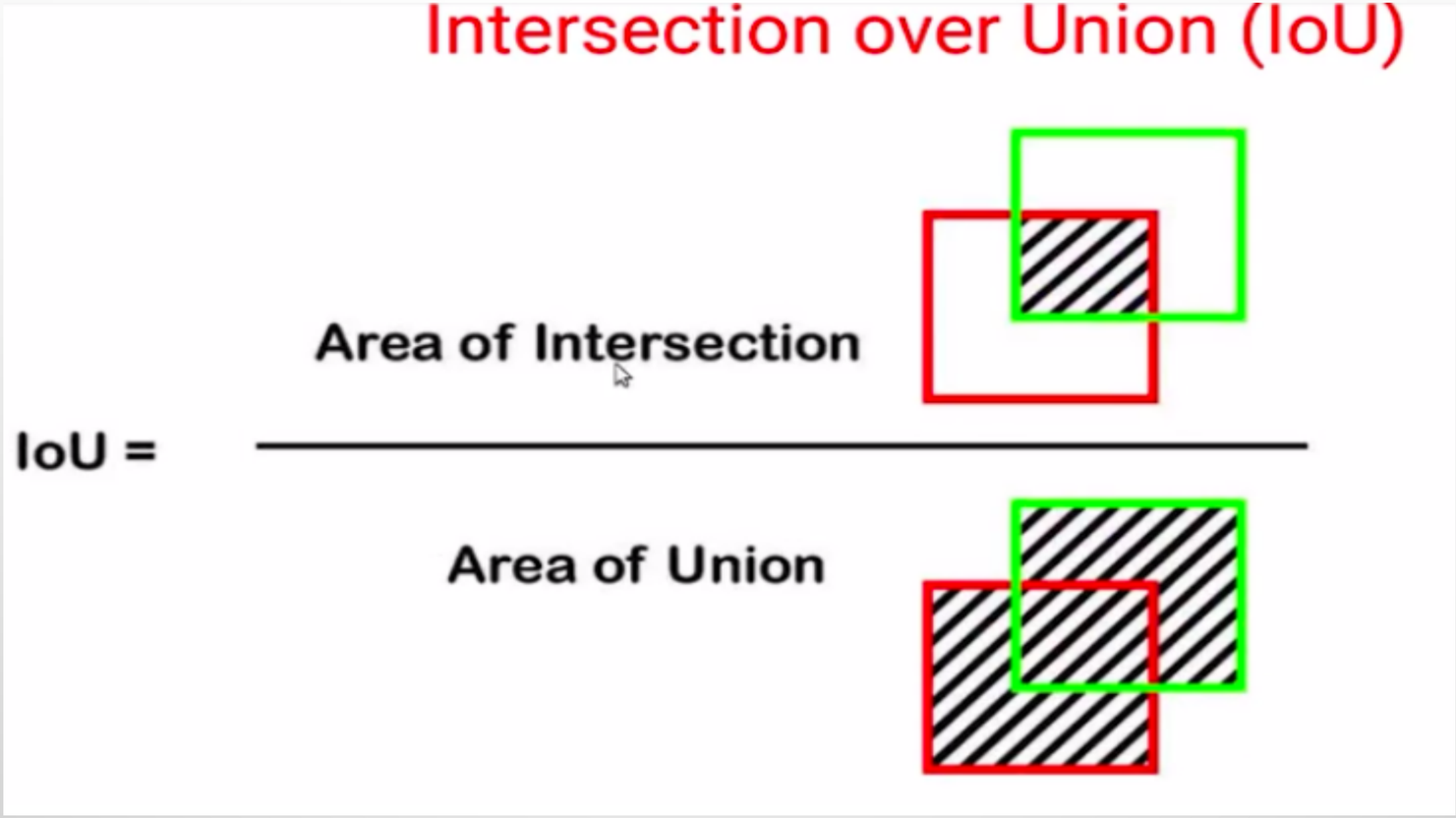
$$IoU = \frac{\text{Area of Intersection}}{\text{Area of Union}}$$



Predicted Bounding Boxes

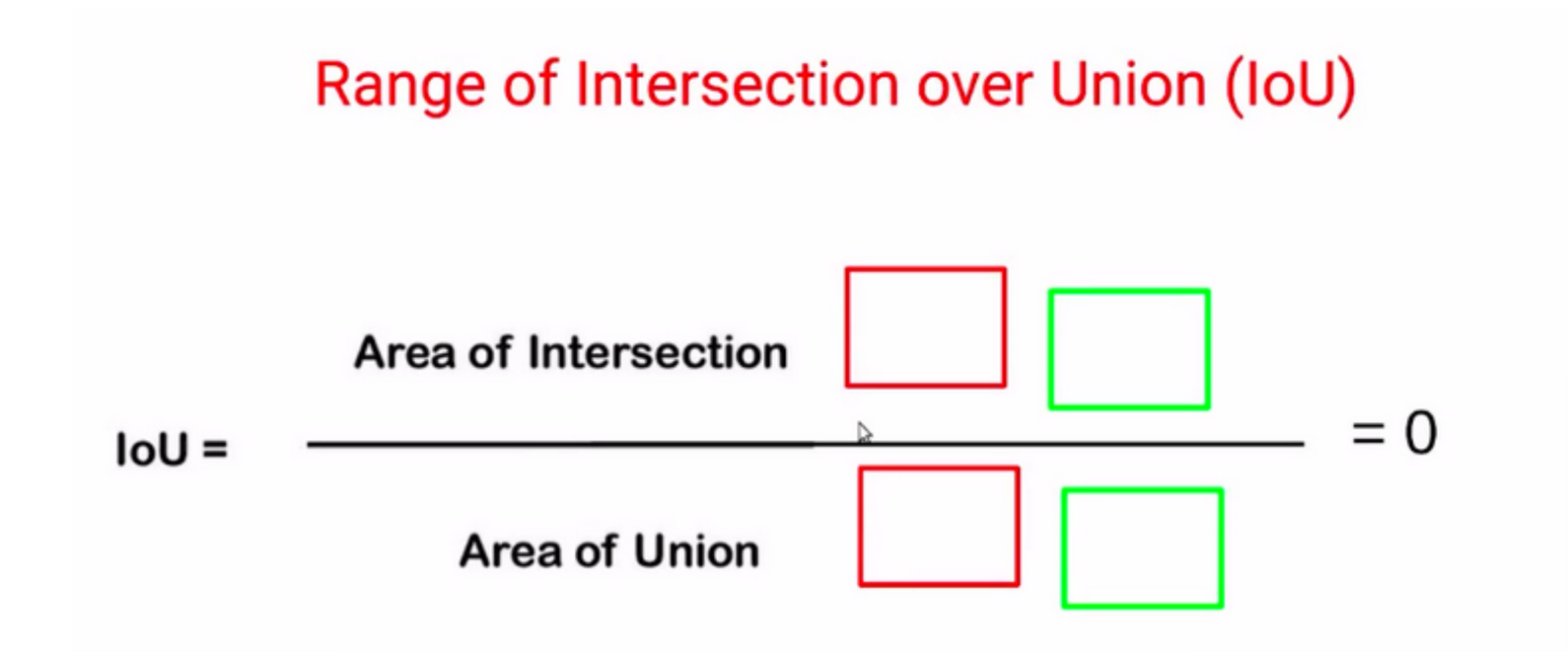
Higher this area of union(blue region) we can say that less accurate will be the predicted bounding box, or the particular patch. Now, this is known as intersection over the union(IoU).

Bounding Box Evaluation – Intersection over Union (IoU)



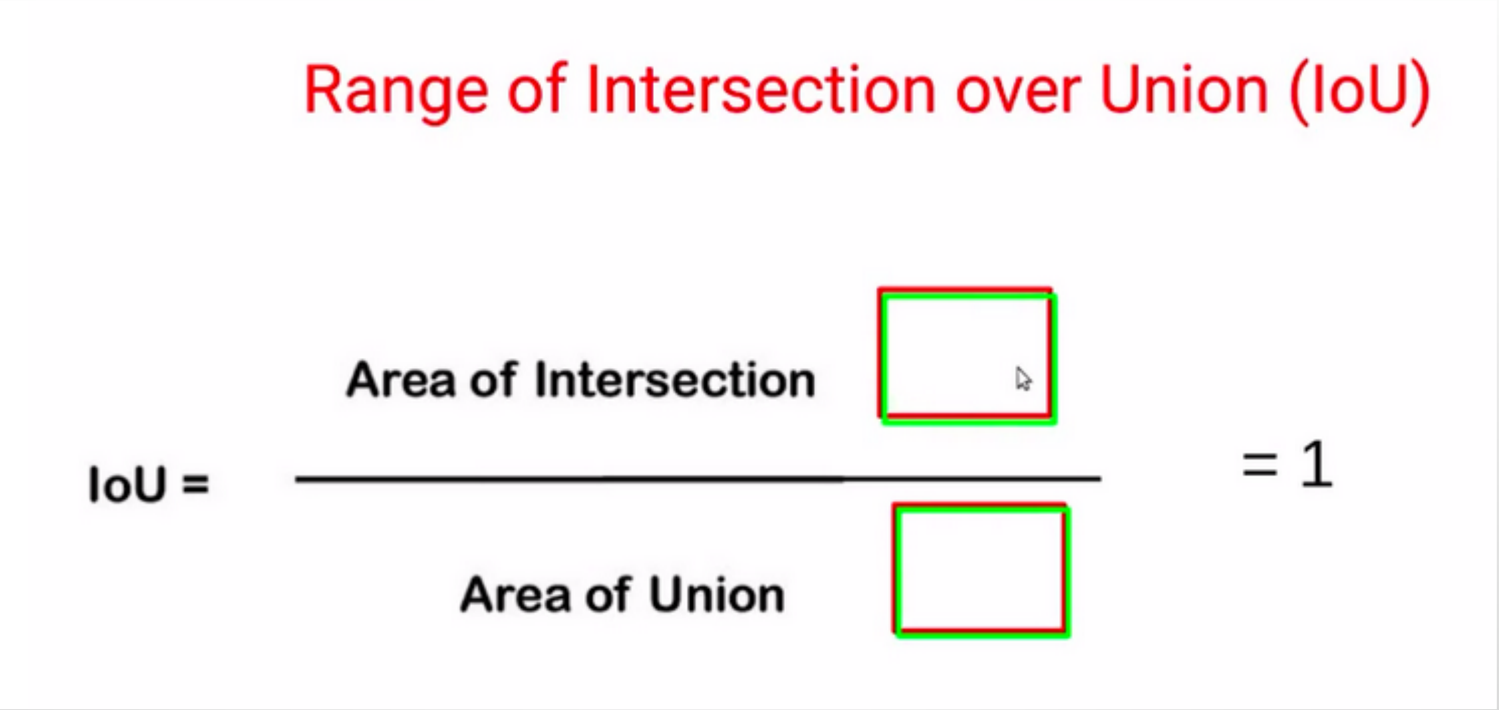
The formula for the intersection over union, which is the area of the intersection divided by the area of union.

Range of Intersection over Union (IoU)



In case we have our actual bounding box and predicted bounding box, and both of these have no overlap at all, in that case, the area of the intersection will be zero, whereas the area of union will be the sum of the area of this patch. So, overall the IoU would be zero.

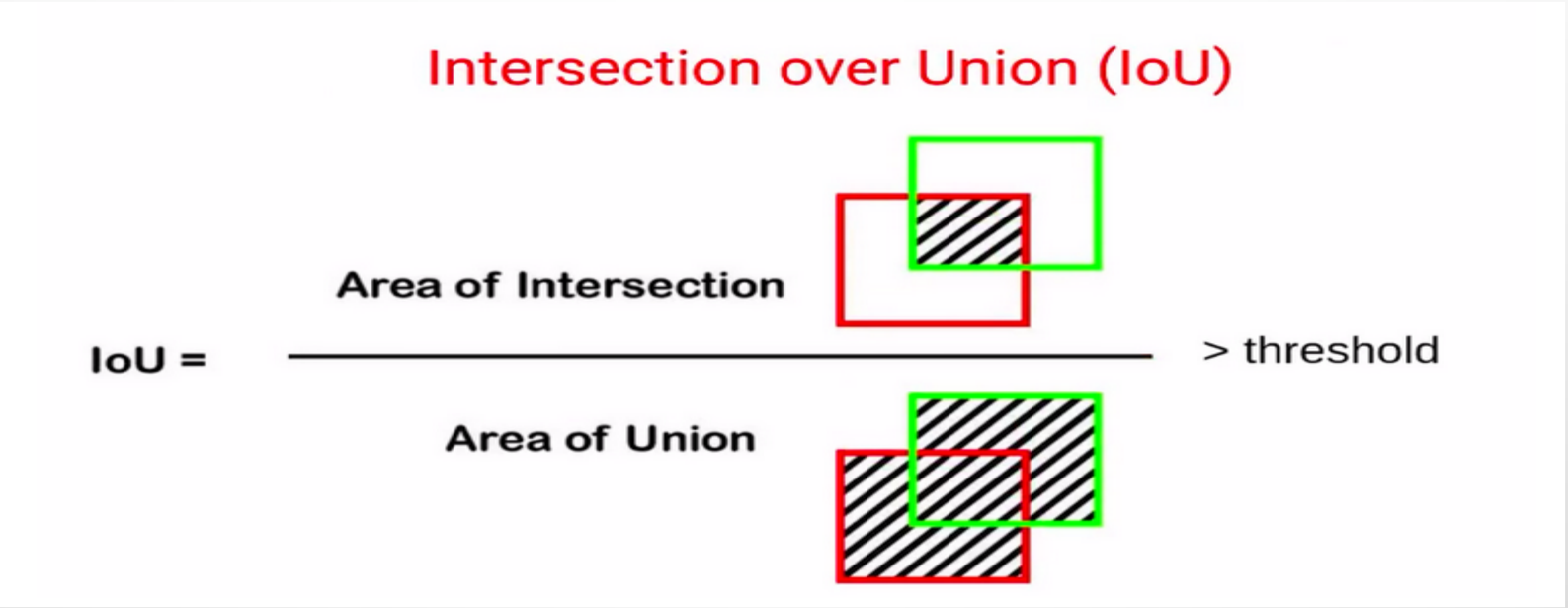
Range of Intersection over Union (IoU)



Scenario:- 2

Another possible scenario could be when both the predicted bounding box and the actual bounding box completely overlap.

Range of Intersection over Union (IoU)

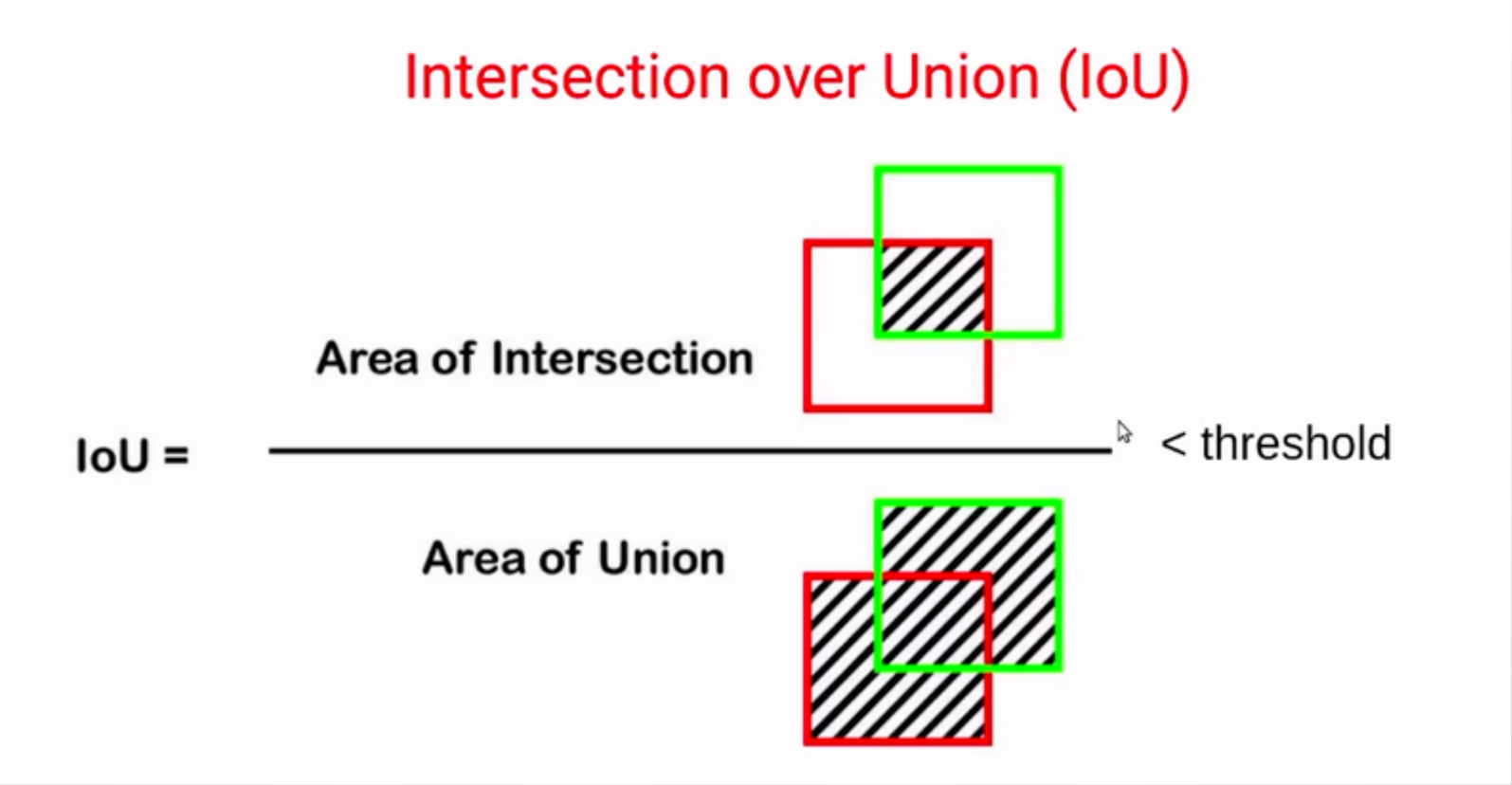


In that case, the area of the intersection will be equal to this overlap, and the area of union will also be the same. Since the numerator and the denominator would be the same in this case, the IoU would be 1.

So, basically, the range of IoU or intersection over union is between 0 and 1.

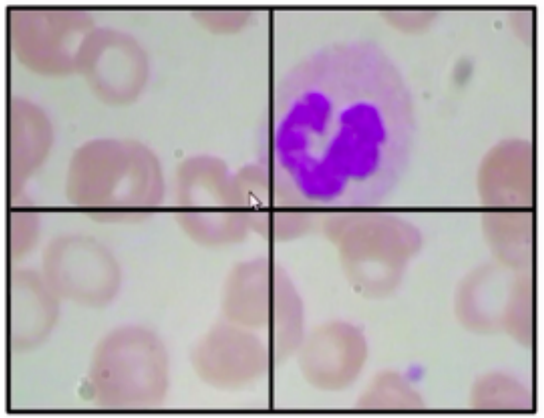
Now we often consider a threshold, in order to identify if the predicted bounding box is the right prediction. So let's say if the IoU is greater than a threshold which can be, let's say 0.5 or 0.6. In that case, we will consider that the actual bounding box and the predicted bounding box are quite similar.

Range of Intersection over Union (IoU)



Whereas if the IoU is less than a particular threshold, we'll say that The predicted bounding box is nothing close to the actual bounding box.
Whereas if the IoU is less than a particular threshold, we'll say that the predicted bounding box is nothing close to the actual bounding box.

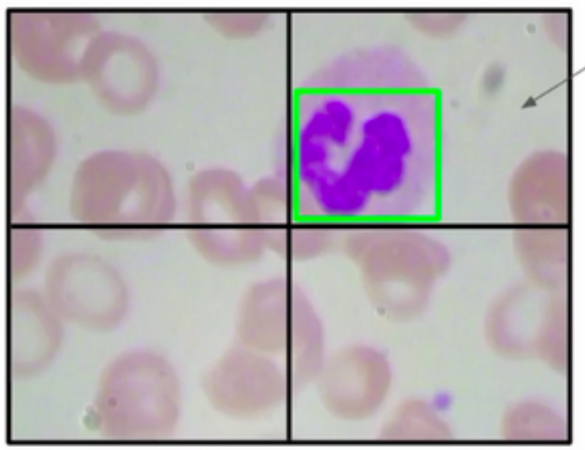
Range of Intersection over Union (IoU)



filename	patch	xmin	xmax	ymin	ymax	WBC (1/0)
1.jpg	1	0	320	0	240	0
1.jpg	2	320	640	0	240	1
1.jpg	3	0	320	240	480	0
1.jpg	4	320	640	240	480	0

Lets consider the intersection over union for a particular threshold. Let's say if the Iou value is greater than 0.5, we'll classify that the particular patch has a WBC and if the IoU is less than this particular threshold we can say that the particular patch does not have the WBC.

Range of Intersection over Union (IoU)



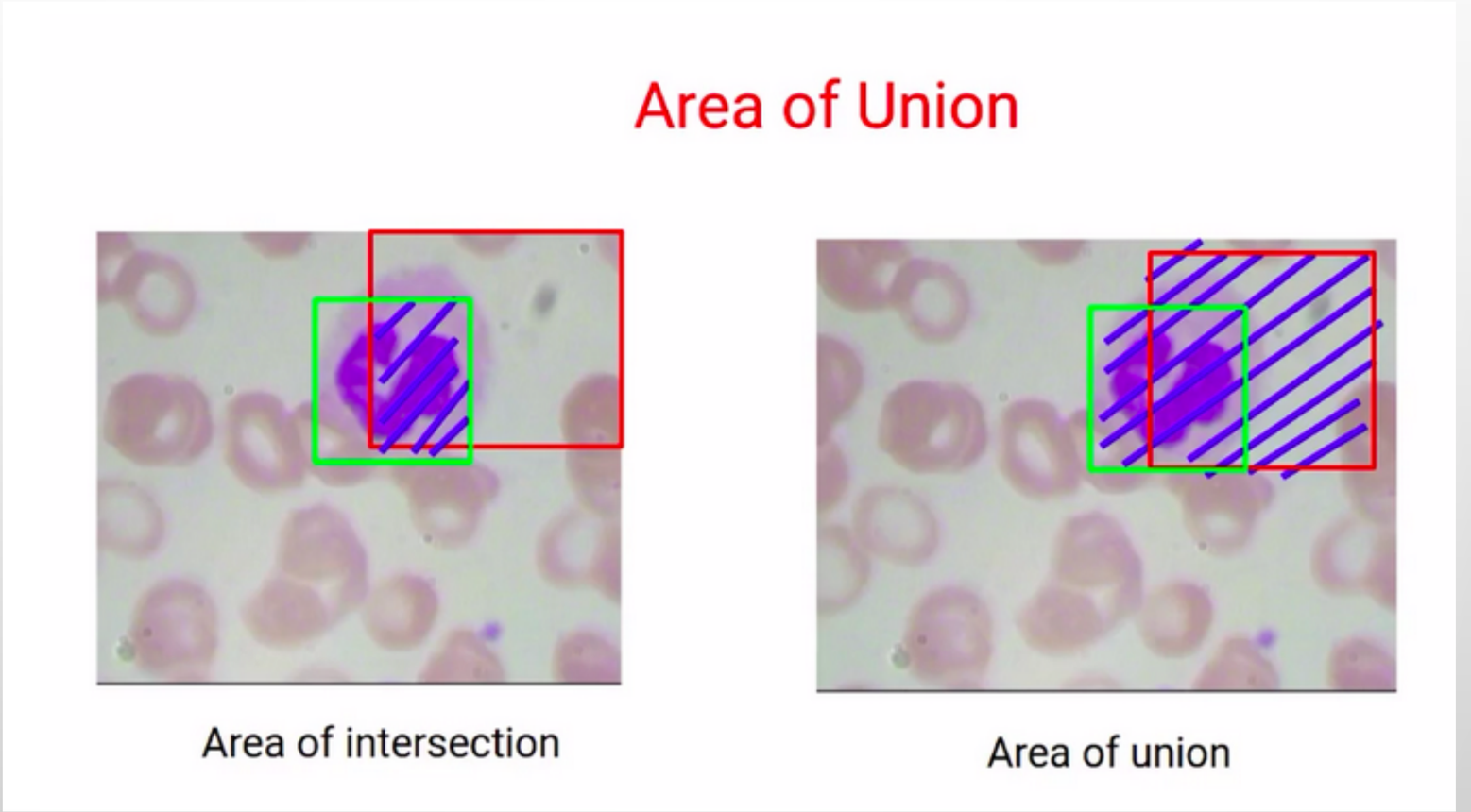
filename	patch	xmin	xmax	ymin	ymax	WBC (1/0)
1.jpg	1	0	320	0	240	0
1.jpg	2	320	640	0	240	1
1.jpg	3	0	320	240	480	0
1.jpg	4	320	640	240	480	0

We are obviously free to set this threshold at our own end.

Now apart from using “IoU”, It Can be Used as:-

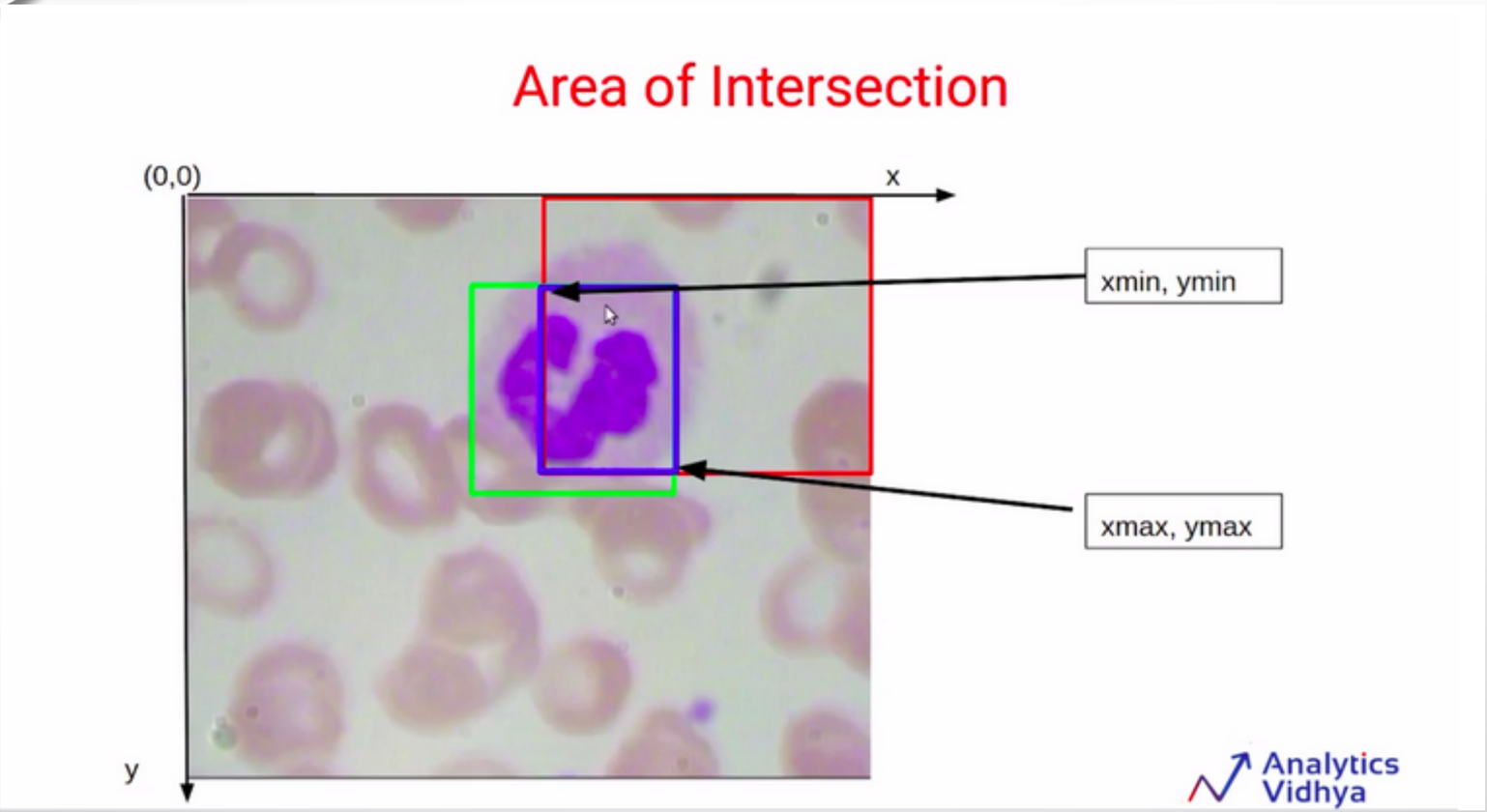
For selecting the best bounding box
As an evaluation Metric

Calculation IOU



we need the area of this blue box. And we can calculate that using the coordinates for this blue box.

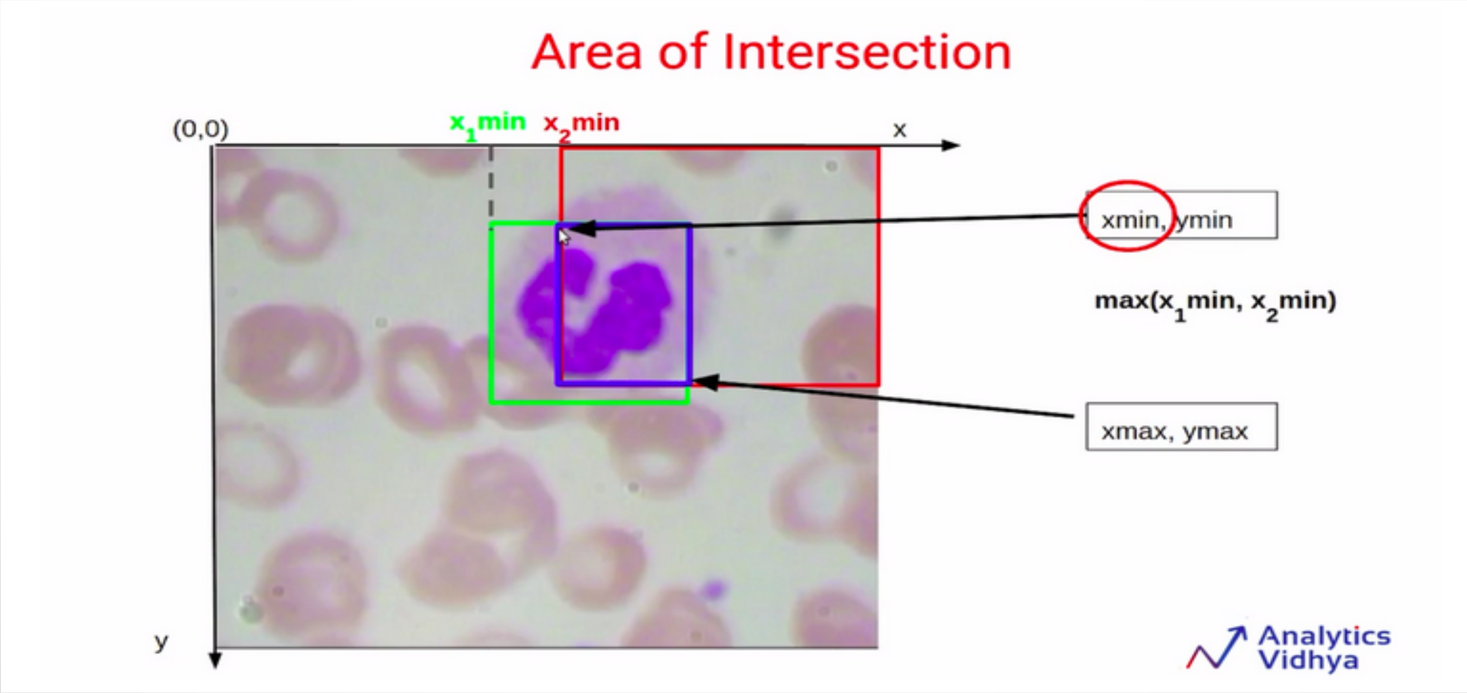
Calculation IOU



So the coordinates will be Xmin, Ymin, Xmax and, Ymax using these coordinates values will be easily able to calculate the area of intersection. So let's focus on determining the value of Xmin here.

In order to find out the value of Xmin, we are going to use the Xmin values for these two bounding boxes, which are represented as X1min and X2min.

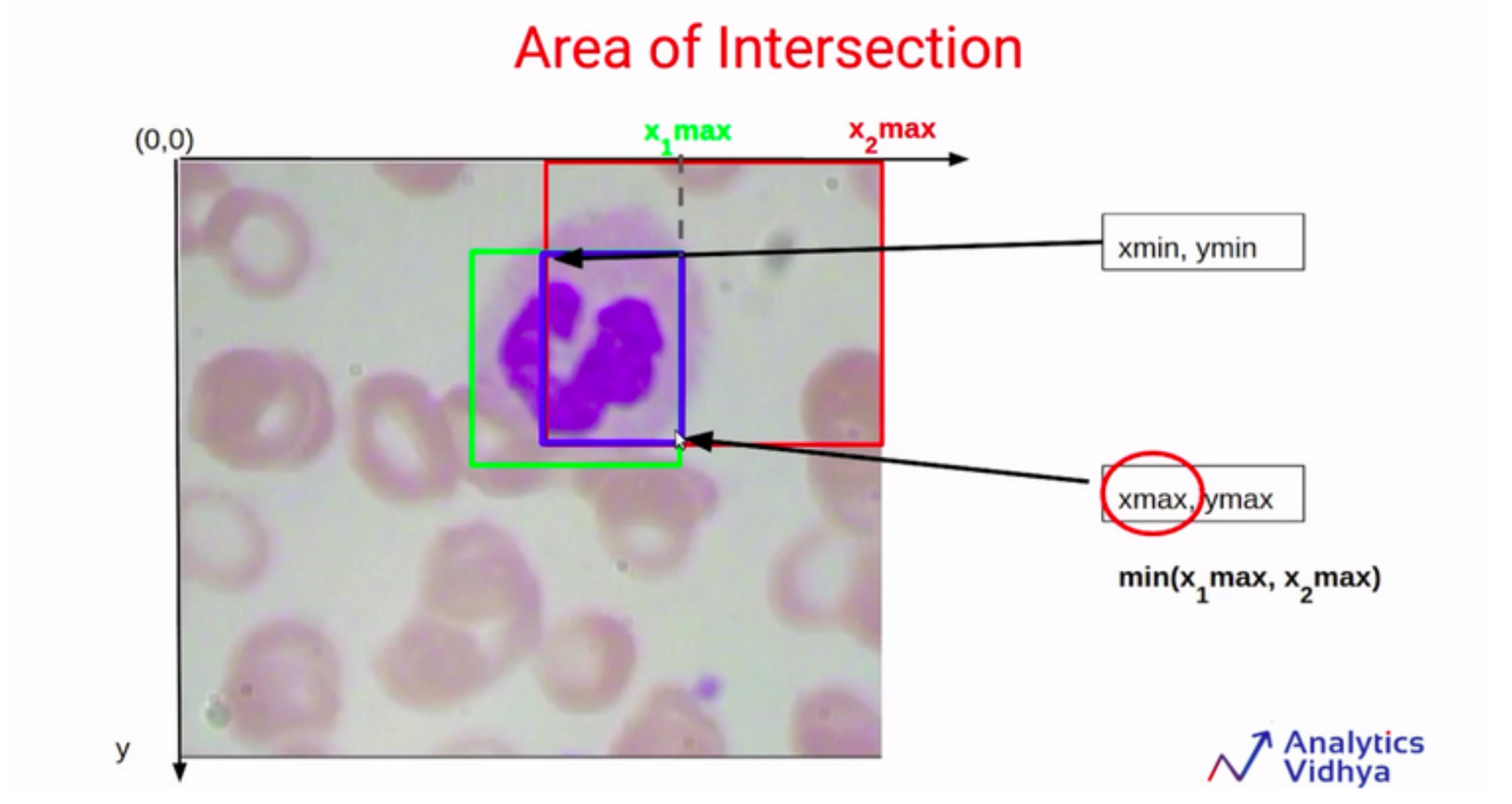
Calculation IOU



In the diagram, the Xmin for this blue bounding box is simply equivalent to X2min. We can also say that the Xmin for this blue box will always be the maximum value out of these two values X1min and X2min.

Similarly, in order to find out the value of Xmax for this blue bounding box, we are going to compare the values X1max and X2max. We can see that the Xmax for this blue bounding box is equivalent to X1max. It can also be written as the minimum of X1max and X2max.

Calculation IOU



Similarly in order to find out the value for Ymin and Ymax. We are going to compare the Y1min and Y2min, and Y1max and Y2max. The value of Ymin will simply be the maximum of Y1 minimum and Y2 minimum which you can see here.

Various Object Detection Methods

R-CNN

Region-based Convolutional Neural Networks (CNNs) that use a two-stage approach to detect objects.

YOLO

You Only Look Once, a real-time object detection system that uses a single neural network to predict bounding boxes and class probabilities.

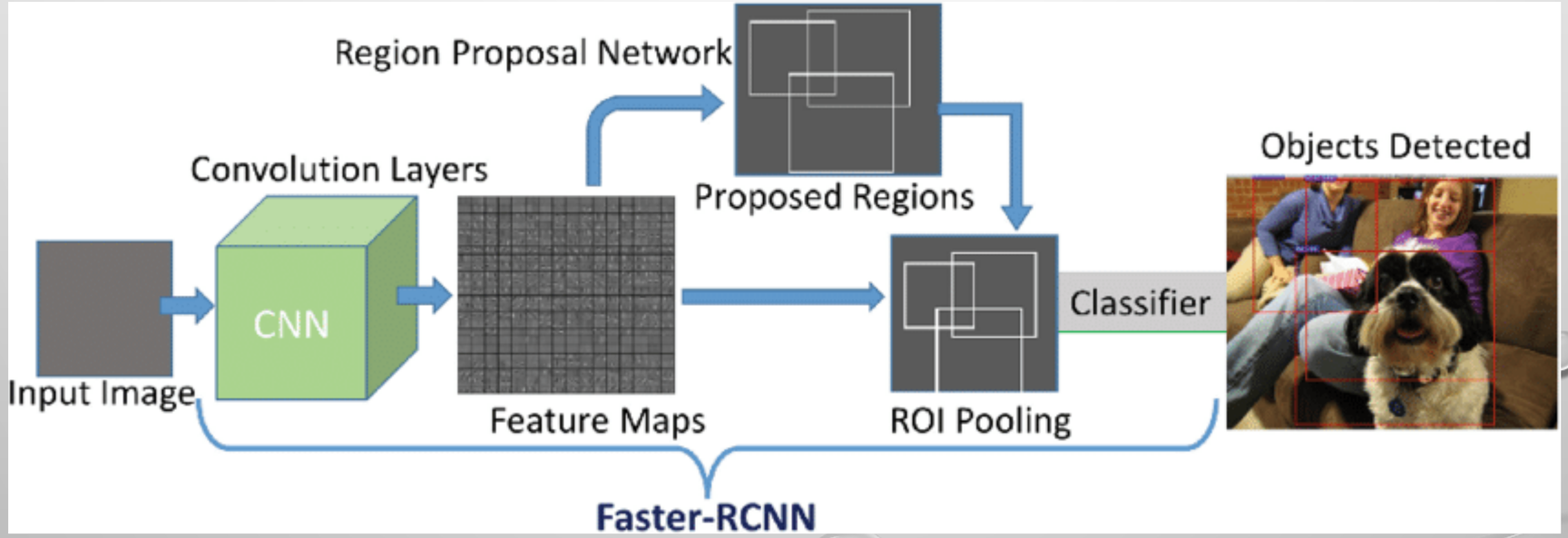
SSD

Single Shot MultiBox Detector, a real-time object detection system that uses a single neural network to predict bounding boxes and class probabilities.

R-CNN Object Detection Method

R-CNN, or Region-based Convolutional Neural Network, is a popular object detection method in computer vision. It uses a two-stage approach, first generating region proposals and then classifying and refining them using a convolutional neural network.

Faster R-CNN is an improved version of R-CNN that uses a single neural network to perform both region proposal generation and feature extraction. This results in faster processing times and improved accuracy.



R-CNN Object Detection - Challenges

1. Computational Cost: R-CNN is a computationally expensive method, requiring significant processing power and memory. This can limit its use in real-time applications.

2. Training Time: Training an R-CNN model can take a significant amount of time, especially for large datasets and complex models.

3. Overfitting: R-CNN models can suffer from overfitting, especially when dealing with small datasets or noisy data. This can result in poor performance on unseen data.

4. Occlusion: R-CNN models can struggle with occlusion, where objects are partially hidden by other objects or background. This can result in missed detections or incorrect classifications.

5. Object Detection Limitations: R-CNN is primarily designed for object detection, and may not be as effective for other computer vision tasks such as segmentation or tracking.

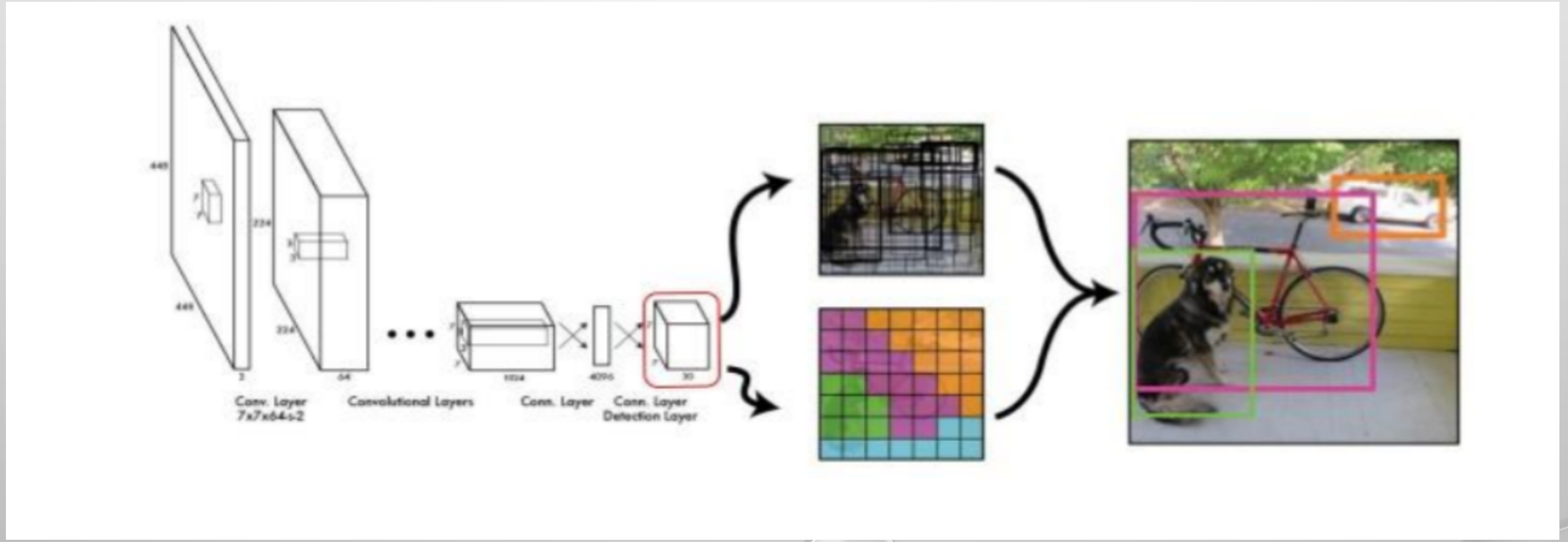
R-CNN Object Detection

ADVANTAGES	LIMITATIONS
• R-CNN is highly accurate and efficient, with state-of-the-art performance on object detection benchmarks. • It can handle complex objects and occlusions, making it suitable for real-world applications.	• R-CNN requires a large amount of training data and computational resources, making it less accessible for researchers and practitioners with limited resources. • It can be computationally expensive and time-consuming, especially for large images or videos.

YOLO Object Detection Method

YOLO (You Only Look Once) is a popular object detection algorithm used in computer vision applications. It was first introduced in 2016 by Joseph Redmon and Ali Farhadi, and has since become one of the most widely used object detection algorithms in the field.

YOLO uses a single neural network to predict bounding boxes and class probabilities directly from full images, without the need for region proposal networks (RPNs) or other preprocessing steps. This makes it faster and more efficient than other object detection algorithms.



YOLO Object Detection Applications

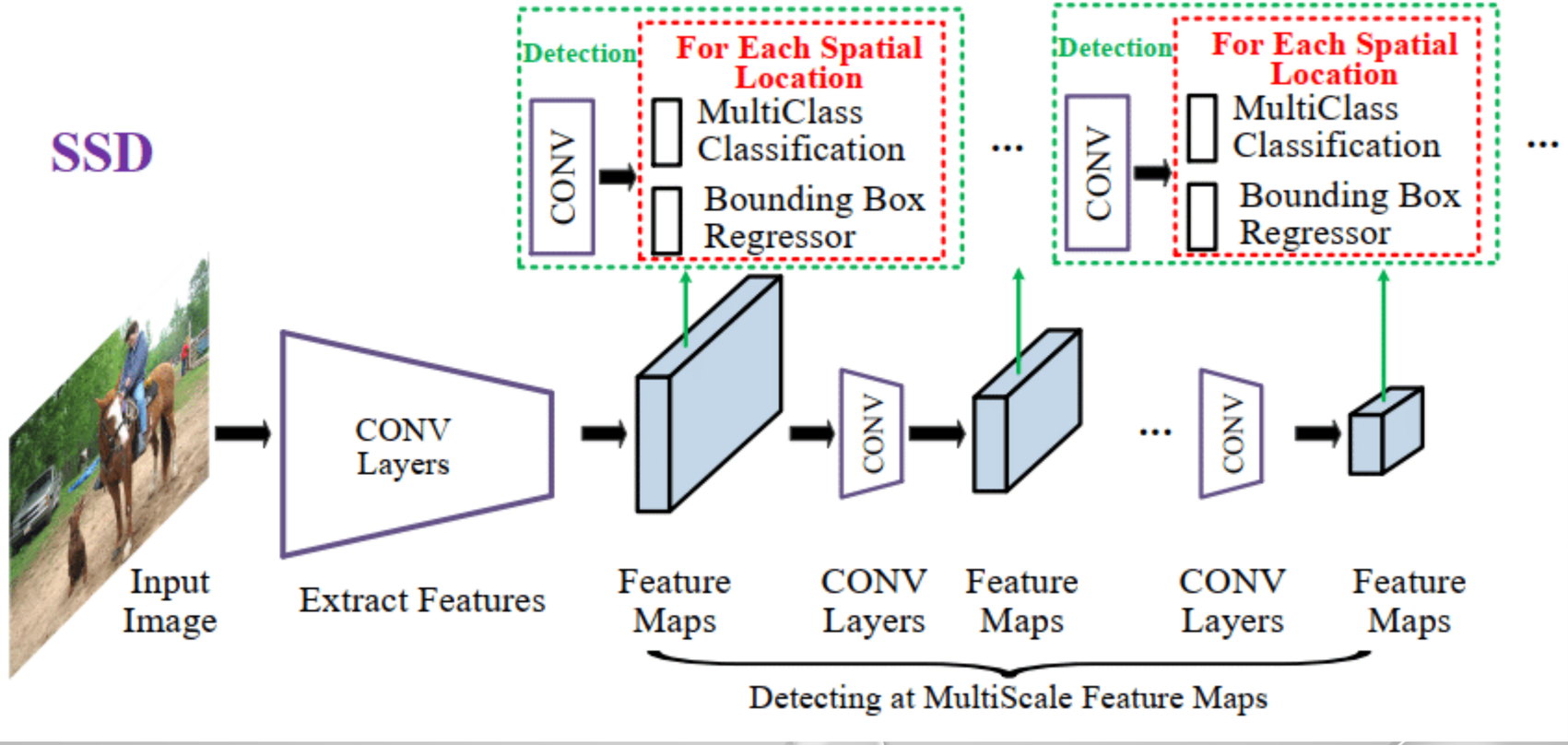
- Autonomous vehicles: YOLO can be used to detect objects such as pedestrians, cars, and traffic signs, which is crucial for self-driving cars.
- Surveillance: YOLO can be used to detect and track people and vehicles in real-time, making it useful for security applications.
- Retail: YOLO can be used to detect and track products on shelves, making it useful for inventory management and customer analytics.
- Healthcare: YOLO can be used to detect and track medical equipment and patients, making it useful for hospital management and patient monitoring.

YOLO Object Detection Challenges

Real-time Performance	Occlusion Detection	Classification Accuracy
One of the biggest challenges in YOLO object detection is achieving real-time performance. This is because YOLO requires a large amount of computation to process images quickly, which can be a challenge for hardware with limited resources.	Another challenge in YOLO object detection is occlusion detection. This occurs when objects are partially hidden by other objects or by the background, making it difficult for the algorithm to accurately detect them.	YOLO object detection also faces challenges in classification accuracy. This is because the algorithm relies on a single neural network to classify objects, which can lead to errors in detection and classification.

SSD Object Detection Method

The SSD architecture consists of three main components: the backbone network, the feature pyramid network, and the detection network. The backbone network is responsible for extracting features from the input image, while the feature pyramid network generates a hierarchy of feature maps at different scales. The detection network then uses these feature maps to predict bounding boxes and class probabilities for each object in the image.



SSD Object Detection Applications

Autonomous Driving	Surveillance Systems	Robotics
SSD object detection is used in autonomous driving to detect and classify objects in the vehicle's surroundings, such as pedestrians, other vehicles, and road signs.	SSD object detection is used in surveillance systems to detect and track people and vehicles, and to identify potential threats or suspicious behavior.	SSD object detection is used in robotics to enable robots to interact with their environment and perform tasks, such as picking up objects or navigating through a space.

SSD Object Detection Challenges

1. Computational Cost: SSD object detection requires significant computational resources, which can be a challenge for real-time applications.

2. Occlusion: SSD object detection can struggle with occluded objects, which can lead to false negatives or missed detections.

3. Lighting Conditions: SSD object detection can be affected by changes in lighting conditions, which can impact detection accuracy.

4. Object Size: SSD object detection can struggle with detecting small objects, which can limit its usefulness in certain applications.

Object Detection

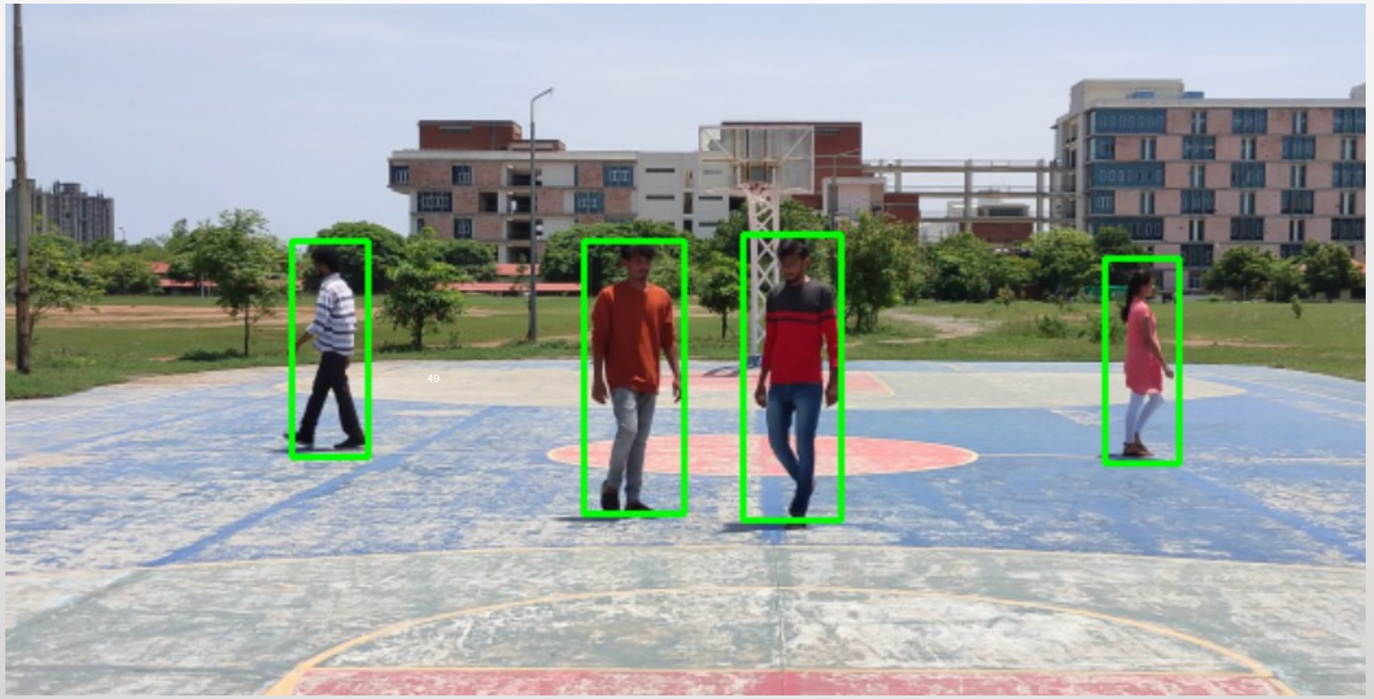


Figure: Object Detection

Object Tracking

Figure: Object Tracking

Applications and Challenges of Object Detection

Applications

- Autonomous driving
- Detecting areal objects
- Live object tracking
- Under water cable tracking imaging system
- Marine biological research
- Satellite imaging system
- Industrial inspection system

Challenges

- Pre-scale resolution training
- Class imbalance in foreground background
- Smaller object detection
- Need of large dataset and computing power
- Error in localization during prediction
- Occlusion
- Multiple instances

Running a pre-built face recognition model.

- # Step 1: Import necessary libraries
- import face_recognition
- import matplotlib.pyplot as plt
- import matplotlib.patches as patches
- from PIL import Image
- # Step 2: Load the image with faces
- image_path = 'path/to/your/image.jpg'
- image = face_recognition.load_image_file(image_path)
- # Step 3: Find face locations in the image
- face_locations = face_recognition.face_locations(image)

Running a pre-built face recognition model.

```
# Step 4: Display the original image with face rectangles
plt.imshow(image)
ax = plt.gca()
for face_location in face_locations:
    # Extract face coordinates
    top, right, bottom, left = face_location
    width = right - left
    height = bottom - top
    # Create a rectangle patch
    rect = patches.Rectangle((left, top), width, height, linewidth=2, edgecolor='r', facecolor='none')
    # Add the rectangle to the Axes
    ax.add_patch(rect)
# Step 5: Show the plot
plt.show()
```

Running a pre-built face recognition model.

- **Explanation**
- Step 1: Import necessary libraries, including face_recognition, matplotlib, and PIL (Pillow for image processing).
- Step 2: Load the image using face_recognition.load_image_file.
- Step 3: Use face_recognition.face_locations to find the locations of faces in the image. This function returns a list of tuples, where each tuple represents the top, right, bottom, and left coordinates of a detected face.
- Step 4: Display the original image with rectangles around the detected faces. This step uses matplotlib to draw rectangles around the faces.
- Step 5: Show the plot using plt.show().