Object Detection

1.
Question 1
Check all the techniques that can be used to improve the accuracy of detecting objects and encapsulating them entirely within a single bounding box.

1 / 1 point

☑

Increase the size of the bounding box until the object fits entirely in it.

Correct

Correct! That is one of the simplest techniques.

☑

Use Selective Search technique

Correct

Correct! It is an advanced technique, and faster than a naive approach.

☐

Scale down the image and then detect the object within it using the bounding box

2.
Question 2
Check all that are true for *Selective Search*.

1 / 1 point

☐

The biggest bounding box detected of the smaller objects in the end becomes the final bounding box around the identified object.

☑

It tries to identify larger objects by grouping together initially identified smaller objects.

Correct

Correct!

☑

Image segmentation is used in this technique

Correct

Correct! It is used to identify smaller objects.

3.
Question 3
The technique of selecting the best bounding box based on the highest intersection over union (IOU) between the true label and several predicted bounding boxes is called non-maximum _____ (NMS). (Hint: it is a one word answer)
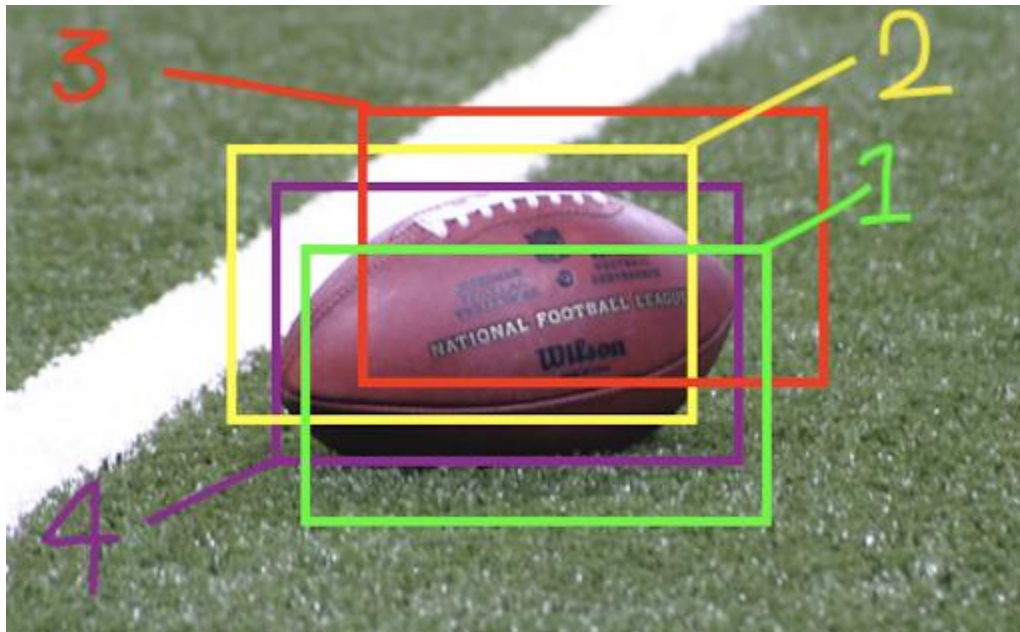
1 / 1 point
suppression
Correct

Correct!

4.
Question 4
Consider the following image, according to the NMS technique which coloured bounding box will be eventually selected as the best bounding box around the football?

1 / 1 point
◯

Green (# 1)

○

Red (# 3)

○

Yellow (# 2)

◉

Purple (# 4)

Correct

Correct! As this bounding box encapsulates the maximum area of the object.

5.
Question 5
One of the differences between R-CNN and Fast R-CNN is that, *Fast R-CNN proposes regions of interest to the input image (generates), whereas in R-CNN regions of interest are expected to be an input (as opposed to generating them) to the model.*

1 / 1 point

○

True

◉

False

Correct

Correct! R-CNN generates regions of interest to the input image, whereas in Fast R-CNN regions of interest are an input (as opposed to generating them).

6.
Question 6
Consider the following code and check all that are true.

```
viz_utils.visualize_boxes_and_labels_on_image_array(
      image_np_with_detections[0],
      result['detection_boxes'][0],
      (result['detection_classes'][0] + label_id_offset).astype(int),
      result['detection_scores'][0],
      category_index,
      use_normalized_coordinates=True,
      min_score_thresh=.40,
      )
```

1 / 1 point

☑

*label_id_offset*is an adjustment in case the 'detection classes' starting index and actual starting index have an offset between them.

Correct

Correct!

☑

*min_score_thresh*is used to leave out object labels and their bounding boxes if their score falls below the set threshold.

Correct

Correct!

☐

image_np_with_detections[0] is a numpy array containing the image, and 0 index shows there are multiple input images being passed to this function.

☐

Setting *use_normalized_coordinates=True* indicates that your bounding box coordinates are not normalized, so you want them to be normalized.

7.
Question 7
The following code initializes a model and restores pre-trained weights, *detection_model,* using the .config file method

```
configs = config_util.get_configs_from_pipeline_file("xyz.config")

model_config = configs['model']
model_config.ssd.num_classes = num_classes
model_config.ssd.freeze_batchnorm = True

detection_model = model_builder.build(
        model_config=model_config, is_training=True)
```

1 / 1 point

○

True

◉

False

Correct

Correct! The code here only initializes a new model architecture with "empty" weights and does not restore pre-trained weights.

8.
Question 8
Which of the following is the correct syntax to print a list of your trainable variables in a model ?

1 / 1 point

○

for varName in myModel.trainables:

print(varName.name)

○

for varName in myModel.trainableVariables:

print(varName.name)

◉

for varName in myModel.trainable_variables:

print(varName.name)

○

for varName in myModel.Variables:

print(varName.name)

Correct

Correct!