**What is Object Detection?**

Object detection is an important computer vision function, responsible for detecting instances of visual objects of certain categories (such as people, plants, or cars). It helps to identify and localize information in digital images that can be used to count and tracks objects in various scenes with labelling and precision.

**Why Object Detection is Important?**

Object detection is one of the important and fundamental problem in field of computer vision that has received great attention from businesses in recent years. The major objective of businesses is to develop computationally efficient models and techniques that extract most of basic information from images or videos needed to build visual real-world applications. These visual applications draw a bounding box around detected objects which allow to locate position of desired objects in images or how they move through a given scene. Some real-world applications using object detection are autonomous driving, robot vision, video surveillance, pedestrian detection, crowd counting, face and text detection system.

**Benefits of Object Detection in Businesses**

**Security**

This Technology is being widely used for surveillance to detect offenders or any other doubtful activities with surveillance cameras. Object Recognition can locate instances present in images or videos that can help to identify suspicious items such as weapons, bags, and vehicle plates. Small or big businesses can use such object recognition technology to identify suspicious activity within their company area and receive instant notification. It also helps to detect people in dangerous or restricted areas or to automate inspection tasks at remote locations.

**Image search**

Object detection is widely being used in e-commerce is image search. With this feature customer can find and locate product easily. One of the best examples of using image search feature is Google Lens. It enables the user to point their phone at somethings such as jacket, shoes, furniture or anything else. The Lens will recognize the object and find similar looking objects along with buying links and suggestions. Another benefit of object recognition is e-commerce product recommendation. When the system identifies an image, it can independently suggest related products to the users, increasing the chances for successful purchase completion that led to grow business.

**Challenges of Object Detection**

Different object detection tasks have completely different objectives and constraints due to which their difficulties vary from each other. Some common challenges for real time visual tasks are to detect objects having multiple viewpoints, illuminations, object rotation, scale changes (e.g., small objects), accurate object localization and speed up of detection, etc. To effectively determine object recognition business are using modern deep learning approaches that depends on supervision training. Deep learning method helps with overcoming limitations like occlusion [partly hidden objects], complex scenarios [no unicolor background], as well as illumination, shadows, and object localization.

**Working of Object Detection**

The tutorial covers a deep convolutional network Mask R-CNN able to learn robust and high-level feature representations of objects within image. The code for detection is available at Google Colab and utilize TPU. Most of businesses are using this model technique as it is capable of achieving state-of-the-art results.

CELL1:

!git clone https://github.com/tensorflow/tpu/

CELL2:

# Importing Python Libraries

from IPython import display

from PIL import Image

import numpy as np

%tensorflow\_version 1.x

import tensorflow as tf

import sys

sys.path.insert(0, 'tpu/models/official')

sys.path.insert(0, 'tpu/models/official/mask\_rcnn')

import coco\_metric

from mask\_rcnn.object\_detection import visualization\_utils

CELL3:

#COCO Index Mapping. Pretrained CheckPoint of model

ID\_MAPPING = {

    1: 'person',

    2: 'bicycle',

    3: 'car',

    4: 'motorcycle',

    5: 'airplane',

    6: 'bus',

    7: 'train',

    8: 'truck',

    9: 'boat',

    10: 'traffic light',

    11: 'fire hydrant',

    13: 'stop sign',

    14: 'parking meter',

    15: 'bench',

    16: 'bird',

    17: 'cat',

    18: 'dog',

    19: 'horse',

    20: 'sheep',

    21: 'cow',

    22: 'elephant',

    23: 'bear',

    24: 'zebra',

    25: 'giraffe',

    27: 'backpack',

    28: 'umbrella',

    31: 'handbag',

    32: 'tie',

    33: 'suitcase',

    34: 'frisbee',

    35: 'skis',

    36: 'snowboard',

    37: 'sports ball',

    38: 'kite',

    39: 'baseball bat',

    40: 'baseball glove',

    41: 'skateboard',

    42: 'surfboard',

    43: 'tennis racket',

    44: 'bottle',

    46: 'wine glass',

    47: 'cup',

    48: 'fork',

    49: 'knife',

    50: 'spoon',

    51: 'bowl',

    52: 'banana',

    53: 'apple',

    54: 'sandwich',

    55: 'orange',

    56: 'broccoli',

    57: 'carrot',

    58: 'hot dog',

    59: 'pizza',

    60: 'donut',

    61: 'cake',

    62: 'chair',

    63: 'couch',

    64: 'potted plant',

    65: 'bed',

    67: 'dining table',

    70: 'toilet',

    72: 'tv',

    73: 'laptop',

    74: 'mouse',

    75: 'remote',

    76: 'keyboard',

    77: 'cell phone',

    78: 'microwave',

    79: 'oven',

    80: 'toaster',

    81: 'sink',

    82: 'refrigerator',

    84: 'book',

    85: 'clock',

    86: 'vase',

    87: 'scissors',

    88: 'teddy bear',

    89: 'hair drier',

    90: 'toothbrush',

}

category\_index = {k: {'id': k, 'name': ID\_MAPPING[k]} for k in ID\_MAPPING}

CELL4:

#Image for object detection

#!wget https://upload.wikimedia.org/wikipedia/commons/thumb/0/08/Kitano\_Street\_Kobe01s5s4110.jpg/2560px-Kitano\_Street\_Kobe01s5s4110.jpg -O test.jpg

image\_path = '/content/image.jpg'

with open(image\_path, 'rb') as f:

  np\_image\_string = np.array([f.read()])

image = Image.open(image\_path)

width, height = image.size

np\_image = np.array(image.getdata()).reshape(height, width, 3).astype(np.uint8)

display.display(display.Image(image\_path, width=1024))

use\_tpu = True #@param {type:"boolean"}

if use\_tpu:

  import os

  import pprint

  assert 'COLAB\_TPU\_ADDR' in os.environ, 'ERROR: Not connected to a TPU runtime; please see the first cell in this notebook for instructions!'

  TPU\_ADDRESS = 'grpc://' + os.environ['COLAB\_TPU\_ADDR']

  print('TPU address is', TPU\_ADDRESS)

  session = tf.Session(TPU\_ADDRESS, graph=tf.Graph())

  print('TPU devices:')

  pprint.pprint(session.list\_devices())

else:

  session = tf.Session(graph=tf.Graph())

saved\_model\_dir = 'gs://cloud-tpu-checkpoints/mask-rcnn/1555659850' #@param {type:"string"}

\_ = tf.saved\_model.loader.load(session, ['serve'], saved\_model\_dir)

CELL5:

num\_detections, detection\_boxes, detection\_classes, detection\_scores, detection\_masks, image\_info = session.run(

    ['NumDetections:0', 'DetectionBoxes:0', 'DetectionClasses:0', 'DetectionScores:0', 'DetectionMasks:0', 'ImageInfo:0'],

    feed\_dict={'Placeholder:0': np\_image\_string})

num\_detections = np.squeeze(num\_detections.astype(np.int32), axis=(0,))

detection\_boxes = np.squeeze(detection\_boxes \* image\_info[0, 2], axis=(0,))[0:num\_detections]

detection\_scores = np.squeeze(detection\_scores, axis=(0,))[0:num\_detections]

detection\_classes = np.squeeze(detection\_classes.astype(np.int32), axis=(0,))[0:num\_detections]

instance\_masks = np.squeeze(detection\_masks, axis=(0,))[0:num\_detections]

ymin, xmin, ymax, xmax = np.split(detection\_boxes, 4, axis=-1)

processed\_boxes = np.concatenate([xmin, ymin, xmax - xmin, ymax - ymin], axis=-1)

segmentations = coco\_metric.generate\_segmentation\_from\_masks(instance\_masks, processed\_boxes, height, width)

CELL6:

max\_boxes\_to\_draw = 50   #@param {type:"integer"}

min\_score\_thresh = 0.1    #@param {type:"slider", min:0, max:1, step:0.01}

image\_with\_detections = visualization\_utils.visualize\_boxes\_and\_labels\_on\_image\_array(

    np\_image,

    detection\_boxes,

    detection\_classes,

    detection\_scores,

    category\_index,

    instance\_masks=segmentations,

    use\_normalized\_coordinates=False,

    max\_boxes\_to\_draw=max\_boxes\_to\_draw,

    min\_score\_thresh=min\_score\_thresh)

output\_image\_path = 'test\_results.jpg'

Image.fromarray(image\_with\_detections.astype(np.uint8)).save(output\_image\_path)

display.display(display.Image(output\_image\_path, width=1024))

**Conclusion**

Object Detection technology has bought many benefits to the companies and has help them to deliver best experience and service to the users.