

# Object Detection Tutorial

martes, 29 de enero de 2019 15:58

**Última Fecha Actualización: 31/01/2019**

## Recursos:

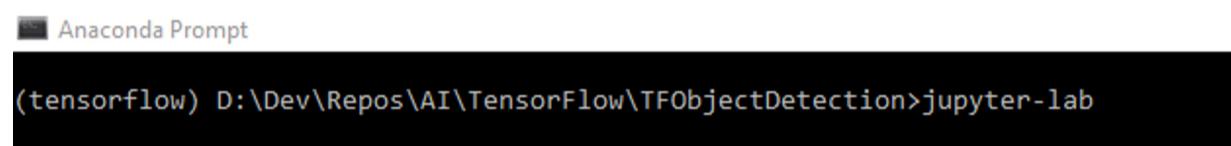
- Quick Start: Jupyter notebook for off-the-shelf inference: [https://github.com/tensorflow/models/tree/master/research/object\\_detection](https://github.com/tensorflow/models/tree/master/research/object_detection)
- Object Detection Tutorial: [https://github.com/tensorflow/models/blob/master/research/object\\_detection/object\\_detection\\_tutorial.ipynb](https://github.com/tensorflow/models/blob/master/research/object_detection/object_detection_tutorial.ipynb)

## Objetivos de la PoC:

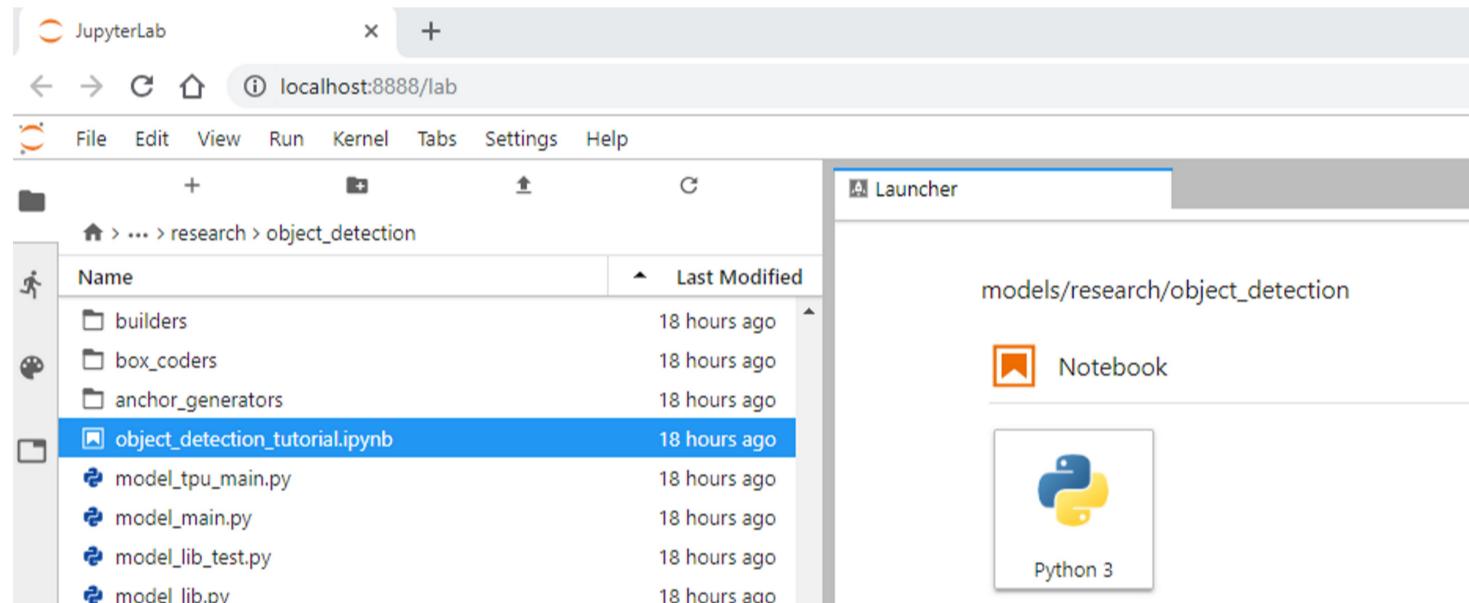
- Revisar el paso a paso de la inferencia de detección de objetos (*inference object detection*).
- Se muestra el proceso de usar un modelo pre-entrenado para detectar objetos en una imagen.
- Cualquier modelo exportado usando la herramienta *export\_inference\_graph.py* puede ser cargado simplemente cambiando la variable *PATH\_TO\_FROZEN\_GRAPH* para apuntar al nuevo archivo .pb.
- **Por defecto se usa en el notebook un modelo "SSD con MobileNet". El modelo usado es "ssd\_mobilenet\_v1\_coco\_2017\_11\_17".**
- Se pueden correr otros modelos "out-of-the-box" con diferentes velocidades y precisiones revisando el listado en *detection\_model\_zoo* ([https://github.com/tensorflow/models/blob/master/research/object\\_detection/g3doc/detection\\_model\\_zoo.md](https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md))

## Detalles PoC:

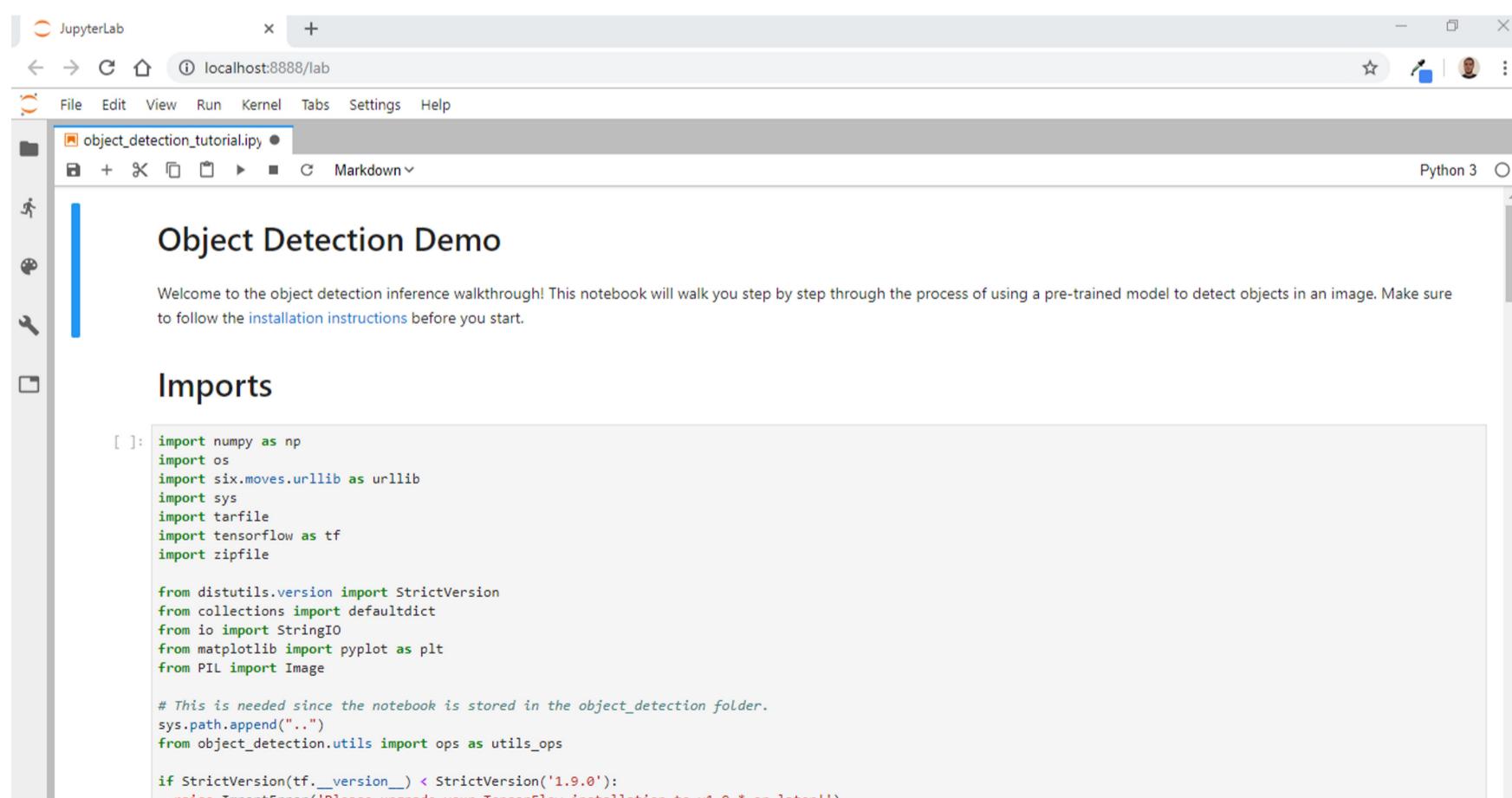
- El "Quick Start" es un Jupyter Notebook oficial de TensorFlow. Por lo tanto, para ejecutarlo abrimos **Anaconda Prompt** y ejecutamos el comando "*jupyter-notebook o jupyter-lab*":



- Para abrir el archivo del Notebook navegamos a la ruta "\models\research\object\_detection\" donde se encuentra el archivo "object\_detection\_tutorial.ipynb":



- Abrimos el archivo "object\_detection\_tutorial.ipynb" y seguimos los pasos en el Notebook para empezar a probar **TensorFlow Object Detection API**:



## ISSUES:

- Matplotlib no muestra las imágenes procesadas con la detección de objetos.
- Se realiza investigación pero ninguna de las pruebas realizadas funcionó correctamente.

- El workaround implementado es el siguiente:
  - Código original - Última celda del Notebook:

```

for image_path in TEST_IMAGE_PATHS:
    image = Image.open(image_path)
    # the array based representation of the image will be used later in order to prepare the
    # result image with boxes and labels on it.
    image_np = load_image_into_numpy_array(image)
    # Expand dimensions since the model expects images to have shape: [1, None, None, 3]
    image_np_expanded = np.expand_dims(image_np, axis=0)
    # Actual detection.
    output_dict = run_inference_for_single_image(image_np, detection_graph)

    # Visualization of the results of a detection.
    vis_util.visualize_boxes_and_labels_on_image_array(
        image_np,
        output_dict['detection_boxes'],
        output_dict['detection_classes'],
        output_dict['detection_scores'],
        category_index,
        instance_masks=output_dict.get('detection_masks'),
        use_normalized_coordinates=True,
        line_thickness=8)

    plt.figure(figsize=IMAGE_SIZE)
    plt.imshow(image_np)
  
```

- Código modificado: Se añade código para guardar las imágenes procesadas en una carpeta:

```

counter = 1

for image_path in TEST_IMAGE_PATHS:
    image = Image.open(image_path)
    # the array based representation of the image will be used later in order to prepare the
    # result image with boxes and labels on it.
    image_np = load_image_into_numpy_array(image)
    # Expand dimensions since the model expects images to have shape: [1, None, None, 3]
    image_np_expanded = np.expand_dims(image_np, axis=0)
    # Actual detection.
    output_dict = run_inference_for_single_image(image_np, detection_graph)

    # Visualization of the results of a detection.
    vis_util.visualize_boxes_and_labels_on_image_array(
        image_np,
        output_dict['detection_boxes'],
        output_dict['detection_classes'],
        output_dict['detection_scores'],
        category_index,
        instance_masks=output_dict.get('detection_masks'),
        use_normalized_coordinates=True,
        line_thickness=8)

    plt.figure(figsize=IMAGE_SIZE)
    plt.title(f"Image_{counter}")
    plt.imshow(image_np)
    plt.savefig(f"outputs\image_{counter}.png")

    counter = counter + 1
  
```

**Nota:** La carpeta "outputs" se debe crear manualmente.

- El resultado en la carpeta "outputs" es el siguiente:

