物件辨識

- 使用Tensorflow Objection Detection API 進行物件辨識
- Tensorflow 2.0
- Tensorflow Objection Detection API

In [1]:
```
import os
from os import walk
from os.path import join
import pathlib
import tensorflow as tf
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
import warnings
import time
```

In [2]:
```
#讀取資料夾內影像檔案路徑
def read_images(foldpath):

    imagePaths =[]

    for root, dirs, files in walk(foldpath):
        for f in files:
            fullpath = join(root, f)
            print(fullpath)
            imagePaths.append(fullpath)

    return imagePaths

IMAGE_PATHS = read_images('E:/Code/Anaconda_Python/Tensorflow/data/objectImg')
```

```
E:/Code/Anaconda_Python/Tensorflow/data/objectImg\image1.jpg
E:/Code/Anaconda_Python/Tensorflow/data/objectImg\image2.jpg
E:/Code/Anaconda_Python/Tensorflow/data/objectImg\image3.jpg
```

In [3]:
```
# 下載並解壓縮model

def downloadModel(model_name):
    url = 'http://download.tensorflow.org/models/object_detection/'
    modelFile = model_name + '.tar.gz'
    modelDir = tf.keras.utils.get_file(fname=model_name,
                                       origin=url + modelFile,
                                       untar=True)
    return str(modelDir)

modelName = 'ssd_mobilenet_v2_coco_2018_03_29'
pathToModelDir = downloadModel(modelName)
```

In [4]:
```
# 下載標籤檔案
def downloadLabels(filename):
    labelUrl = 'https://raw.githubusercontent.com/tensorflow/models/master/research/
    labelDir = tf.keras.utils.get_file(fname=filename,
                                       origin=labelUrl + filename,
                                       untar=False)
    label_dir = pathlib.Path(labelDir)
    return str(labelDir)
```

```
labelFilename = 'mscoco_label_map.pbtxt'
pathToLabels = downloadLabels(labelFilename)
```

In [5]:

```
from object_detection.utils import label_map_util
from object_detection.utils import visualization_utils as viz_utils

PATH_TO_SAVED_MODEL = pathToModelDir + "/saved_model"

print('Loading model...', end='')
start_time = time.time()

# 讀取model& build the detection function
model = tf.saved_model.load(PATH_TO_SAVED_MODEL)
detect_fn = model.signatures['serving_default']

end_time = time.time()
elapsed_time = end_time - start_time
print('Done! Took {} seconds'.format(elapsed_time))
```

```
d:\anaconda\envs\tensorflow\lib\site-packages\object_detection\utils\visualization_u
tils.py:29: UserWarning:
This call to matplotlib.use() has no effect because the backend has already
been chosen; matplotlib.use() must be called *before* pylab, matplotlib.pyplot,
or matplotlib.backends is imported for the first time.

The backend was *originally* set to 'module://ipykernel.pylab.backend_inline' by the
following code:
  File "d:\anaconda\envs\tensorflow\lib\runpy.py", line 193, in _run_module_as_main
    "__main__", mod_spec)
  File "d:\anaconda\envs\tensorflow\lib\runpy.py", line 85, in _run_code
    exec(code, run_globals)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\ipykernel_launcher.py", line 1
6, in <module>
    app.launch_new_instance()
  File "d:\anaconda\envs\tensorflow\lib\site-packages\traitlets\config\application.p
y", line 845, in launch_instance
    app.start()
  File "d:\anaconda\envs\tensorflow\lib\site-packages\ipykernel\kernelapp.py", line
612, in start
    self.io_loop.start()
  File "d:\anaconda\envs\tensorflow\lib\site-packages\tornado\platform\asyncio.py",
line 199, in start
    self.asyncio_loop.run_forever()
  File "d:\anaconda\envs\tensorflow\lib\asyncio\base_events.py", line 541, in run_fo
rever
    self._run_once()
  File "d:\anaconda\envs\tensorflow\lib\asyncio\base_events.py", line 1786, in _run_
once
    handle._run()
  File "d:\anaconda\envs\tensorflow\lib\asyncio\events.py", line 88, in _run
    self._context.run(self._callback, *self._args)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\tornado\ioloop.py", line 688,
in <lambda>
    lambda f: self._run_callback(functools.partial(callback, future))
  File "d:\anaconda\envs\tensorflow\lib\site-packages\tornado\ioloop.py", line 741,
in _run_callback
    ret = callback()
  File "d:\anaconda\envs\tensorflow\lib\site-packages\tornado\gen.py", line 814, in
inner
    self.ctx_run(self.run)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\tornado\gen.py", line 775, in
run
    yielded = self.gen.send(value)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\ipykernel\kernelbase.py", line
358, in process_one
```

```
    yield gen.maybe_future(dispatch(*args))
  File "d:\anaconda\envs\tensorflow\lib\site-packages\tornado\gen.py", line 234, in
wrapper
    yielded = ctx_run(next, result)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\ipykernel\kernelbase.py", line
261, in dispatch_shell
    yield gen.maybe_future(handler(stream, idents, msg))
  File "d:\anaconda\envs\tensorflow\lib\site-packages\tornado\gen.py", line 234, in
wrapper
    yielded = ctx_run(next, result)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\ipykernel\kernelbase.py", line
538, in execute_request
    user_expressions, allow_stdin,
  File "d:\anaconda\envs\tensorflow\lib\site-packages\tornado\gen.py", line 234, in
wrapper
    yielded = ctx_run(next, result)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\ipykernel\ipkernel.py", line 3
02, in do_execute
    res = shell.run_cell(code, store_history=store_history, silent=silent)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\ipykernel\zmqshell.py", line 5
39, in run_cell
    return super(ZMQInteractiveShell, self).run_cell(*args, **kwargs)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\IPython\core\interactiveshell.
py", line 2899, in run_cell
    self.events.trigger('post_run_cell', result)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\IPython\core\events.py", line
89, in trigger
    func(*args, **kwargs)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\ipykernel\pylab\backend_inlin
e.py", line 216, in configure_once
    activate_matplotlib(backend)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\IPython\core\pylabtools.py", l
ine 322, in activate_matplotlib
    plt.switch_backend(backend)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\matplotlib\pyplot.py", line 23
1, in switch_backend
    matplotlib.use(newbackend, warn=False, force=True)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\matplotlib\__init__.py", line
1422, in use
    reload(sys.modules['matplotlib.backends'])
  File "d:\anaconda\envs\tensorflow\lib\importlib\__init__.py", line 169, in reload
    _bootstrap._exec(spec, module)
  File "d:\anaconda\envs\tensorflow\lib\site-packages\matplotlib\backends\__init__.p
y", line 17, in <module>
    line for line in traceback.format_stack()


  import matplotlib; matplotlib.use('Agg')  # pylint: disable=multiple-statements
Loading model...INFO:tensorflow:Saver not created because there are no variables in
the graph to restore
Done! Took 5.0418946743011475 seconds
```

In [7]:

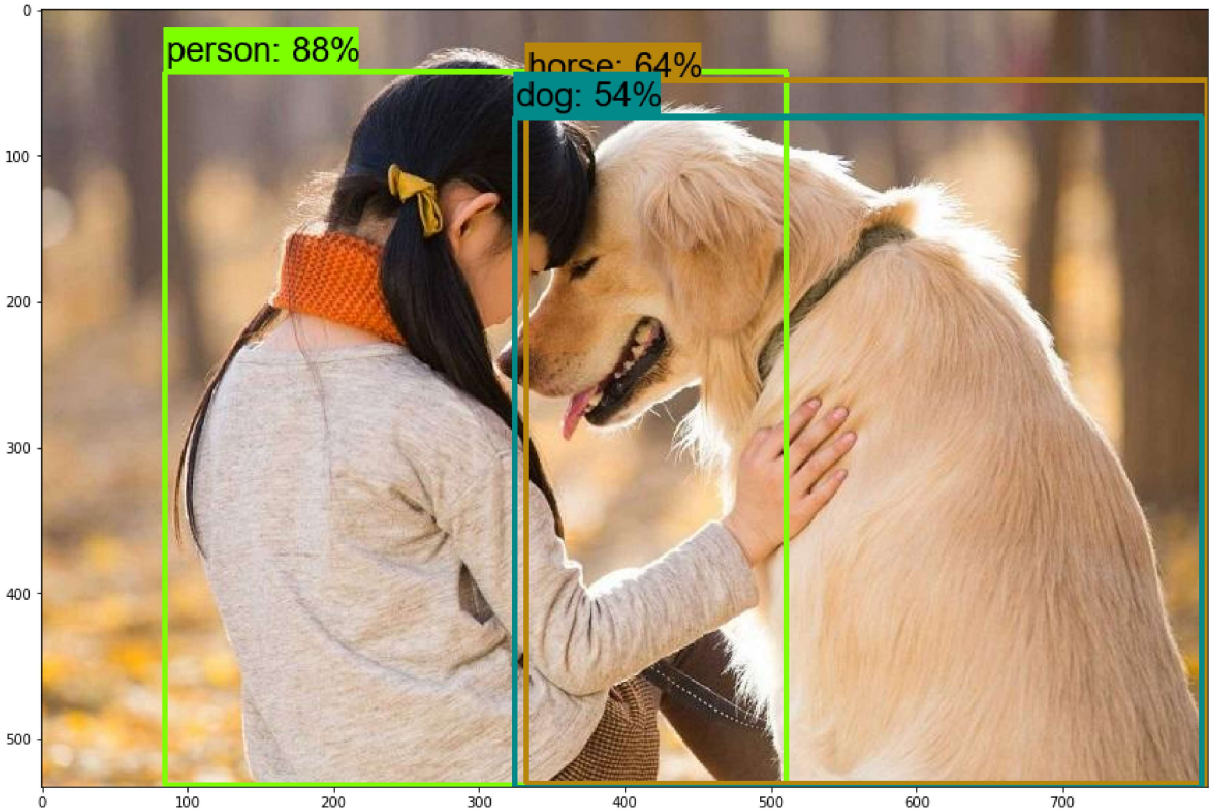
```
categoryIndex = label_map_util.create_category_index_from_labelmap(pathToLabels,
                                                                   use_display_name
```

In [8]:

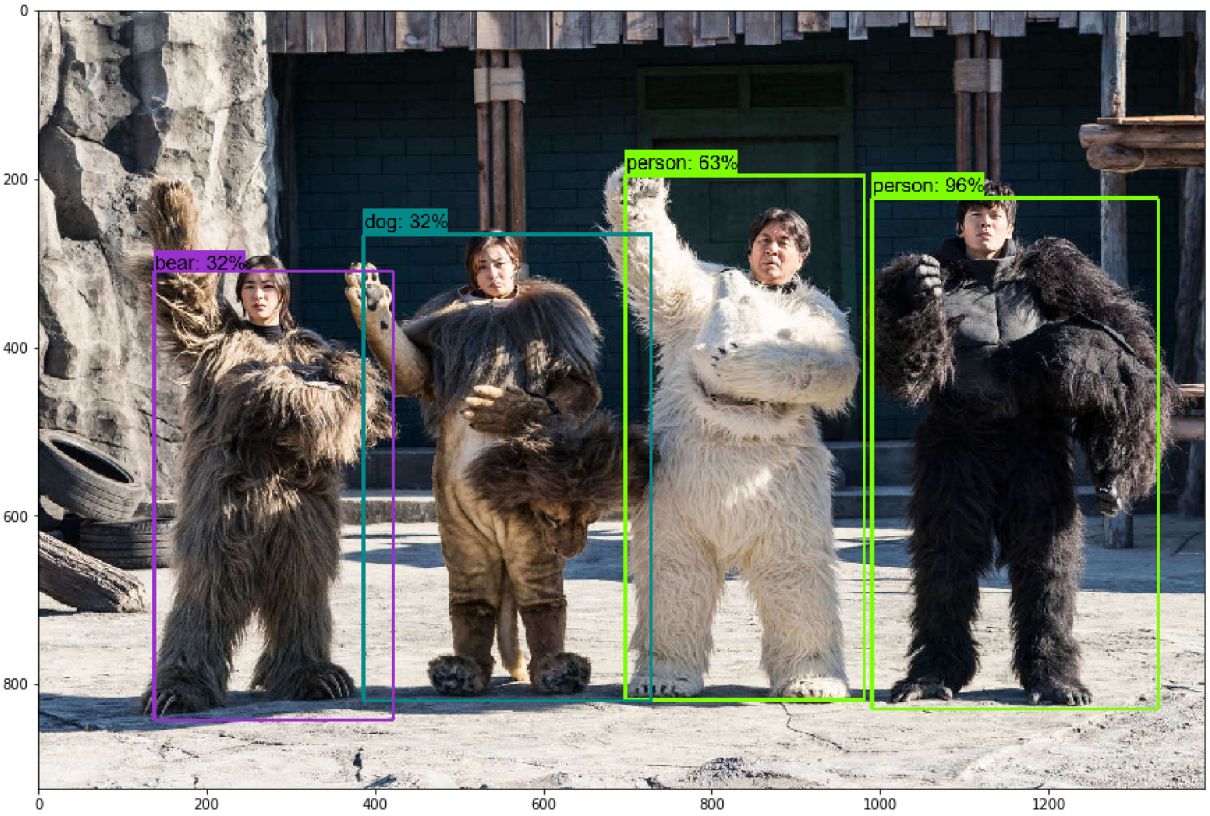
```
def loadImage_to_numpyArray(path):
    return np.array(Image.open(path))

for image_path in IMAGE_PATHS:
    print('Running inference for {}... '.format(image_path), end='')

    image_np =loadImage_to_numpyArray(image_path)
    input_tensor = tf.convert_to_tensor(image_np)
    input_tensor = input_tensor[tf.newaxis, ...]

    detections = detect_fn(input_tensor)
```

```
    num_detections = int(detections.pop('num_detections'))
    detections = {key: value[0, :num_detections].numpy()
                  for key, value in detections.items()}

    detections['num_detections'] = num_detections

    detections['detection_classes'] = detections['detection_classes'].astype(np.int6

    image_np_with_detections = image_np.copy()

    viz_utils.visualize_boxes_and_labels_on_image_array(
          image_np_with_detections,
          detections['detection_boxes'],
          detections['detection_classes'],
          detections['detection_scores'],
          categoryIndex,
          use_normalized_coordinates=True,
          max_boxes_to_draw=200,
          min_score_thresh=.30,
          agnostic_mode=False)

    plt.figure(figsize=(15,15))
    plt.imshow(image_np_with_detections)
    print('Done')
    plt.show()
```

```
Running inference for E:/Code/Anaconda_Python/Tensorflow/data/objectImg\image1.jp
g... Done
```




```
Running inference for E:/Code/Anaconda_Python/Tensorflow/data/objectImg\image2.jp
g... Done
```




```
Running inference for E:/Code/Anaconda_Python/Tensorflow/data/objectImg\image3.jp
g... Done
```

In [ ]:

In [ ]: