Building an Object Detection Model from Scratch in Python

When we're shown an image, our brain instantly recognizes the objects contained in it. On the other hand, it takes a lot of time and training data for a machine to identify these objects. But with the recent advances in hardware and deep learning, this computer vision field has become a whole lot easier and more intuitive.

https://www.analyticsvidhya.com/blog/2018/06/understanding-building-object-detection-model-python/ (https://www.analyticsvidhya.com/blog/2018/06/understanding-building-object-detection-model-python/)
https://www.analyticsvidhya.com/blog/2018/07/top-10-pretrained-models-get-started-deep-learning-part-1-computer-vision/ (https://www.analyticsvidhya.com/blog/2018/07/top-10-pretrained-models-get-started-deep-learning-part-1-computer-vision/)

Manuel Robalinho - 21-10-2018

Step 1: Create an Anaconda environment with python version 3.6.

conda create -n retinanet python=3.6 anaconda

Step 2: Activate the environment and install the necessary packages.

activate retinanet conda install tensorflow numpy scipy opencv pillow matplotlib h5py keras

Step 3: Then install the ImageAI library.

pip install https://github.com/OlafenwaMoses/ImageAI/releases/download/2.0.1/imageai-2.0.1-py3-none-any.whl (https://github.com/OlafenwaMoses/ImageAI/releases/download/2.0.1/imageai-2.0.1-py3-none-any.whl)

Step 4: Now download the pretrained model required to generate predictions.

This model is based on RetinaNet (a subject of a future article). Click on the link to download – RetinaNet Pretrained model https://github.com/fizyr/keras-retinanet/releases (https://github.com/fizyr/keras-retinanet/releases)

Step 5: Copy the downloaded file to your current working folder

Step 6: Download the image from this link. Name the image as image.png

Step 7: Open jupyter notebook (type jupyter notebook in your terminal) and run the following codes:

In [16]:

```
# Load library to ObjectDetection
from imageai.Detection import ObjectDetection
import os

path = 'ml/Object_Detection/'
execution_path = path
#execution_path = os.getcwd()
```

In [ ]:

```
# Load Model

detector = ObjectDetection()
detector.setModelTypeAsRetinaNet()
detector.setModelPath( os.path.join(execution_path , "resnet50_coco_best_v2.1.0.h5"))
detector.loadModel()
custom_objects = detector.CustomObjects(person=True, car=True)  # Detect Persons and cars
```

In [3]:

```
# Test Model with an image
image = "image.png"
detections = detector.detectCustomObjectsFromImage(input_image=os.path.join(execution_path
for eachObject in detections:
   print(eachObject["name"] + " : " + eachObject["percentage_probability"] )
   print("--------------------------------")
```

```
person : 73.20682406425476
--------------------------------
```

In [10]:

```
from PIL import Image
im = Image.open(path+image)
#im.show()
im
```

Out[10]:

In [13]:

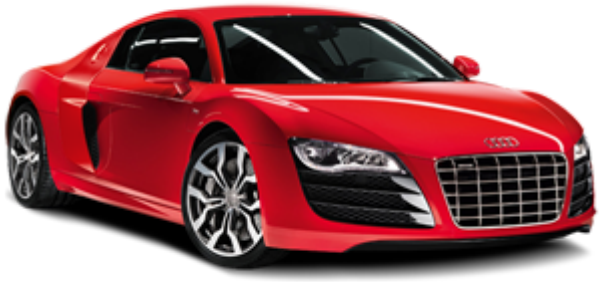
```
# Another Test

image = 'audi_PNG1767.png'
custom_objects = detector.CustomObjects(person=True, car=True)
detections = detector.detectCustomObjectsFromImage(input_image=os.path.join(execution_path

for eachObject in detections:
   print(eachObject["name"] + " : " + eachObject["percentage_probability"] )
   print("--------------------------------")
```

```
car : 76.26890540122986
--------------------------------
car : 94.30564641952515
--------------------------------
car : 70.49705386161804
--------------------------------
```

In [12]:

```
im = Image.open(path+image)
#im.show()
im
```

Out[12]:



In [14]:

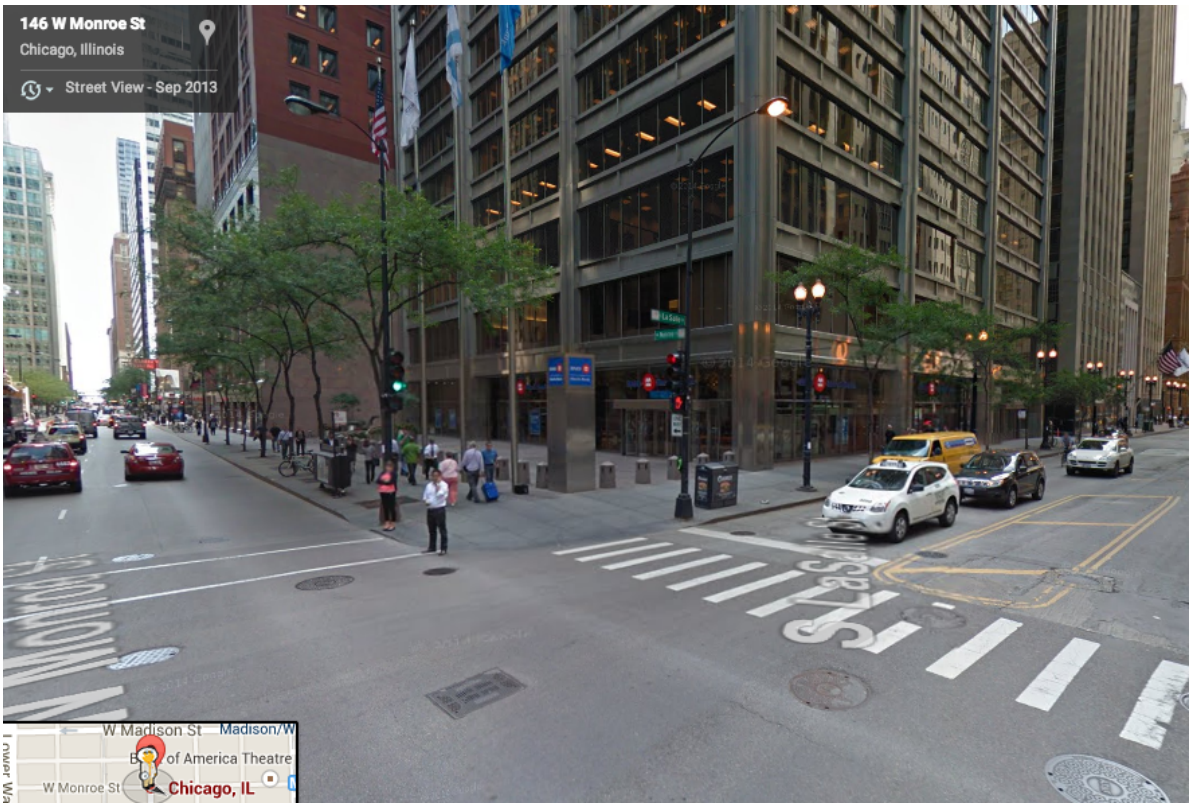
```
# Another Test

image = 'Screen_Shot_2015-01-28_at_2.05.44_PM.0.png'
custom_objects = detector.CustomObjects(person=True, car=True)
detections = detector.detectCustomObjectsFromImage(input_image=os.path.join(execution_path

for eachObject in detections:
   print(eachObject["name"] + " : " + eachObject["percentage_probability"] )
   print("--------------------------------")
```

```
car : 82.46123194694519
--------------------------------
car : 81.64753913879395
--------------------------------
car : 88.45213651657104
--------------------------------
person : 78.99371385574341
--------------------------------
car : 76.92534923553467
--------------------------------
car : 85.50105690956116
--------------------------------
car : 69.24325227737427
--------------------------------
car : 80.25151491165161
--------------------------------
car : 77.58230566978455
--------------------------------
car : 93.2178258895874
--------------------------------
car : 90.0388777256012
--------------------------------
car : 89.04653787612915
--------------------------------
```

In [15]:

```
im = Image.open(path+image)
#im.show()
im
```

Out[15]:



In [ ]: