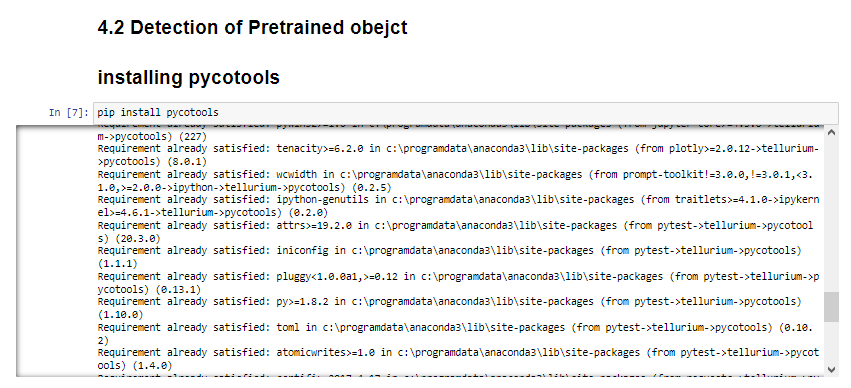
**REVERSE IMAGE SEARCH**



**Figure 1: Installing the necessary pycotools in the Jupyter IDE**

(Source: Acquired from Jupyter notebook)

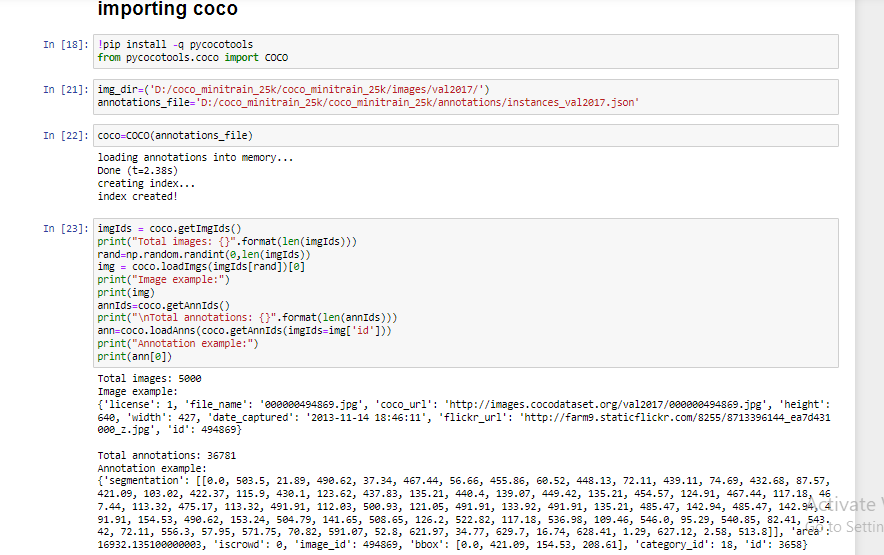
In the above figure, the pycotools library is being installed using the “pip install pycotools” command. Pycotools is a python library for doing and enhancing simulation and modeling.



**Figure 2: Importing all the required library**

(Source: Acquired from Jupyter notebook)

In the above figure, all the important libraries that can be useful for performing the task have been imported accordingly.



**Figure 3: importing cocco**

(Source: Acquired from Jupyter notebook)

In the above image the coco function is being imported from the pycotool library and the coco annotations have been loaded and the images processed. Coco is used to specify large scale detecting, captioning and segmentations datasets.



**Figure 4: Detecting the objects**

(Source: Acquired from Jupyter notebook)

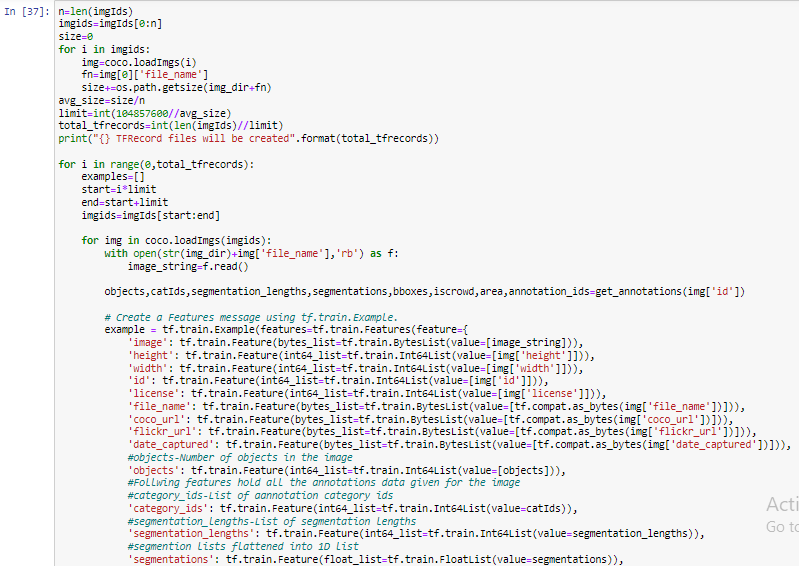
In the above figure the object detection can be seen and it has been performed with the help of using “Python” software.



**Figure 5: Creating tf record output directory**

(Source: Acquired from Jupyter notebook)

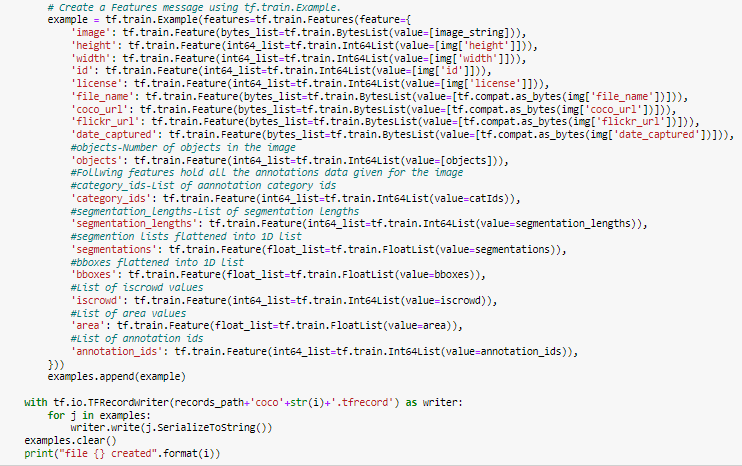
In the above figure the creation of the tensorflow record output directory is being created wherein the files are stored sequentially.



**Figure 6: Creating the feature messages**

(Source: Acquired from Jupyter notebook)

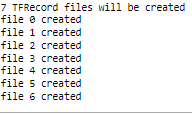
In the above figure the tf records files have been created and the future messages are being created using the tf.train.example.



**Figure 7: Object numbers**

(Source: Acquired from Jupyter notebook)

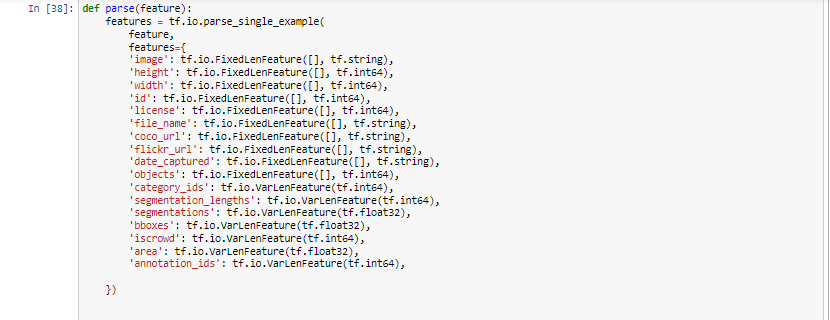
The object numbers of the feature messages that were earlier initiated have been depicted in the above figure.



**Figure 8: TF records file**

(Source: Acquired from Jupyter notebook)

In the above figure there are seven TF records that can be seen which have been generated with the help of using the “python” software.



**Figure 9: Featuring all the segments**

(Source: Acquired from Jupyter notebook)

In the above figure all the segments that have been created using the tf.io.parse are being featured.



**Figure 10: Printing all the images**

(Source: Acquired from Jupyter notebook)

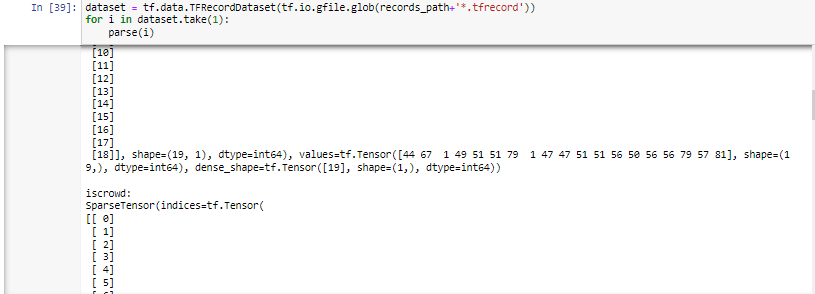
In the above figure all the images have been printed with the help of using the ‘print’ command in the ‘python’ environment.



**Figure 11: Segmentation**

(Source: Acquired from Jupyter notebook)

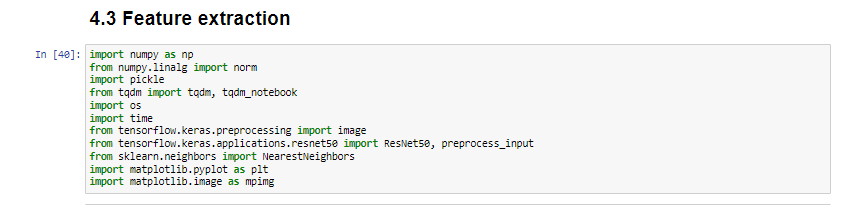
In the above figure the segmentation has been done for the object categorization task. In the case of making the segmentation, the segmentation length has also been set.



**Figure 12: Dataset reading**

(Source: Acquired from Jupyter notebook)

The tf.data.TFRecordDataset along with the path file in the argument has been used to read the dataset in the above figure.



**Figure 13: Importing libraries for feature extraction**

(Source: Acquired from Jupyter notebook)

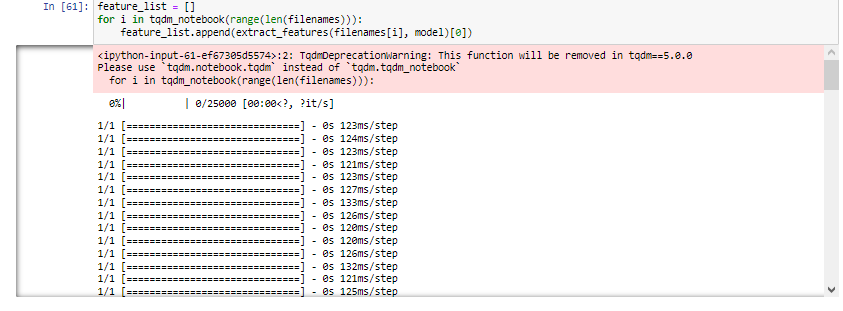
The necessary libraries have been imported as depicted in the above figure for initializing the feature extraction



**Figure 14: Extracting features**

(Source: Acquired from Jupyter notebook)

The features are being extracted in the above figure, the path of the file has been mentioned above.



**Figure 15: Feature list**

(Source: Acquired from Jupyter notebook)

In the above figure there are a lot of feature lists that can be seen and these feature lists have been generated with the help of using the ‘python’ software.



**Figure 16: Similarity search**

(Source: Acquired from Jupyter notebook)

In the above figure a similarity search has been done with the help of using the ‘python’ software.