```
!pip install -U --pre tensorflow=="2.*"
!pip install tf_slim
```

```
Collecting tensorflow==2.*
  Downloading https://files.pythonhosted.org/packages/e4/54/426543e8b616a479c26f47de1ba8edf2a588722955579d44377f803670
     |████████████████████████████████| 394.6MB 26kB/s
Requirement already satisfied, skipping upgrade: typing-extensions~=3.7.4 in /usr/local/lib/python3.6/dist-packages (f
Collecting protobuf~=3.13.0
  Downloading https://files.pythonhosted.org/packages/30/79/510974552cebff2ba04038544799450defe75e96ea5f1675dbf72cc874
     |████████████████████████████████| 1.3MB 41.8MB/s
Requirement already satisfied, skipping upgrade: wheel~=0.35 in /usr/local/lib/python3.6/dist-packages (from tensorflo
Requirement already satisfied, skipping upgrade: wrapt~=1.12.1 in /usr/local/lib/python3.6/dist-packages (from tensorf
Collecting grpcio~=1.32.0
  Downloading https://files.pythonhosted.org/packages/f0/00/b393f5d0e92b37592a41357ea3077010c95400c907f6b9af01f4f6abe1
     |████████████████████████████████| 3.8MB 35.1MB/s
Collecting flatbuffers~=1.12.0
  Downloading https://files.pythonhosted.org/packages/eb/26/712e578c5f14e26ae3314c39a1bdc4eb2ec2f4ddc89b708cf8e0a0d204
Requirement already satisfied, skipping upgrade: astunparse~=1.6.3 in /usr/local/lib/python3.6/dist-packages (from ten
Requirement already satisfied, skipping upgrade: keras-preprocessing~=1.1.2 in /usr/local/lib/python3.6/dist-packages
Collecting numpy~=1.19.2
  Downloading https://files.pythonhosted.org/packages/87/86/753182c9085ba4936c0076269a571613387cdb77ae2bf537448bfd6347
     |████████████████████████████████| 14.5MB 39.9MB/s
Requirement already satisfied, skipping upgrade: termcolor~=1.1.0 in /usr/local/lib/python3.6/dist-packages (from tens
Requirement already satisfied, skipping upgrade: six~=1.15.0 in /usr/local/lib/python3.6/dist-packages (from tensorflo
Requirement already satisfied, skipping upgrade: absl-py~=0.10 in /usr/local/lib/python3.6/dist-packages (from tensorf
Requirement already satisfied, skipping upgrade: gast==0.3.3 in /usr/local/lib/python3.6/dist-packages (from tensorflo
Collecting tensorboard~=2.4
  Downloading https://files.pythonhosted.org/packages/02/83/179c8f76e5716030cc3ee9433721161cfcc1d854e9ba20c9205180bb10
     |████████████████████████████████| 10.6MB 224kB/s
Collecting tensorflow-estimator<2.5.0,>=2.4.0rc0
  Downloading https://files.pythonhosted.org/packages/84/89/5d041ec47d50ab9beeeb14158258d3e4835596cf5dd809e887ead46a7f
     |████████████████████████████████| 471kB 42.3MB/s
Requirement already satisfied, skipping upgrade: h5py~=2.10.0 in /usr/local/lib/python3.6/dist-packages (from tensorfl
Requirement already satisfied, skipping upgrade: google-pasta~=0.2 in /usr/local/lib/python3.6/dist-packages (from ten
Requirement already satisfied, skipping upgrade: opt-einsum~=3.3.0 in /usr/local/lib/python3.6/dist-packages (from ten
Requirement already satisfied, skipping upgrade: setuptools in /usr/local/lib/python3.6/dist-packages (from protobuf~=
Requirement already satisfied, skipping upgrade: google-auth-oauthlib<0.5,>=0.4.1 in /usr/local/lib/python3.6/dist-pac
Requirement already satisfied, skipping upgrade: tensorboard-plugin-wit>=1.6.0 in /usr/local/lib/python3.6/dist-packag
Requirement already satisfied, skipping upgrade: requests<3,>=2.21.0 in /usr/local/lib/python3.6/dist-packages (from t
Requirement already satisfied, skipping upgrade: werkzeug>=0.11.15 in /usr/local/lib/python3.6/dist-packages (from ten
Requirement already satisfied, skipping upgrade: google-auth<2,>=1.6.3 in /usr/local/lib/python3.6/dist-packages (from
Requirement already satisfied, skipping upgrade: markdown>=2.6.8 in /usr/local/lib/python3.6/dist-packages (from tenso
Requirement already satisfied, skipping upgrade: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.6/dist-packages (f
Requirement already satisfied, skipping upgrade: chardet<4,>=3.0.2 in /usr/local/lib/python3.6/dist-packages (from req
Requirement already satisfied, skipping upgrade: idna<3,>=2.5 in /usr/local/lib/python3.6/dist-packages (from requests
Requirement already satisfied, skipping upgrade: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.6/d
```

```
    Requirement already satisfied, skipping upgrade: certifi>=2017.4.17 in /usr/local/lib/python3.6/dist-packages (from re
    Requirement already satisfied, skipping upgrade: rsa<5,>=3.1.4; python_version >= "3" in /usr/local/lib/python3.6/dist
    Requirement already satisfied, skipping upgrade: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.6/dist-packages (from
    Requirement already satisfied, skipping upgrade: cachetools<5.0,>=2.0.0 in /usr/local/lib/python3.6/dist-packages (fro
    Requirement already satisfied, skipping upgrade: importlib-metadata; python_version < "3.8" in /usr/local/lib/python3
```

```
# Python API that helps in loading,parsing and visualizing the annotations in COCO(image dataset)
!pip install pycocotools
```

```
    Requirement already satisfied: pycocotools in /usr/local/lib/python3.6/dist-packages (2.0.2)
    Requirement already satisfied: setuptools>=18.0 in /usr/local/lib/python3.6/dist-packages (from pycocotools) (50.3.2)
    Requirement already satisfied: cython>=0.27.3 in /usr/local/lib/python3.6/dist-packages (from pycocotools) (0.29.21)
    Requirement already satisfied: matplotlib>=2.1.0 in /usr/local/lib/python3.6/dist-packages (from pycocotools) (3.2.2)
    Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=2.1.0-
    Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=2.1.0->pycocot
    Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=2.1.0->py
    Requirement already satisfied: numpy>=1.11 in /usr/local/lib/python3.6/dist-packages (from matplotlib>=2.1.0->pycocoto
    Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.6/dist-packages (fro
    Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from python-dateutil>=2.1->matplotl
```

```
        Uninstalling tensorboard-2.3.0:
```

```
import os
import pathlib
```

```
if "models" in pathlib.Path.cwd().parts:
  while "models" in pathlib.Path.cwd().parts:
    os.chdir('..')
elif not pathlib.Path('models').exists():
  !git clone --depth 1 https://github.com/tensorflow/models
```

```
    Cloning into 'models'...
    remote: Enumerating objects: 2305, done.
    remote: Counting objects: 100% (2305/2305), done.
    remote: Compressing objects: 100% (2000/2000), done.
    remote: Total 2305 (delta 563), reused 941 (delta 282), pack-reused 0
    Receiving objects: 100% (2305/2305), 30.59 MiB | 30.92 MiB/s, done.
    Resolving deltas: 100% (563/563), done.
```

```
%%bash
cd models/research/
protoc object_detection/protos/*.proto --python_out=.
```

```
%%bash
cd models/research
pip install .
```

```
Processing /content/models/research
Requirement already satisfied: Pillow>=1.0 in /usr/local/lib/python3.6/dist-packages (from object-detection==0.1) (7.0
Requirement already satisfied: Matplotlib>=2.1 in /usr/local/lib/python3.6/dist-packages (from object-detection==0.1)
Requirement already satisfied: Cython>=0.28.1 in /usr/local/lib/python3.6/dist-packages (from object-detection==0.1) (
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.6/dist-packages (from Matplotlib>=2.1->object-de
Requirement already satisfied: numpy>=1.11 in /usr/local/lib/python3.6/dist-packages (from Matplotlib>=2.1->object-det
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.6/dist-packages (from Matplotlib>=2.1->obje
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.6/dist-packages (from Matplotlib>=2.1->o
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.6/dist-packages (fro
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-packages (from cycler>=0.10->Matplotlib>=2.1->obje
Building wheels for collected packages: object-detection
  Building wheel for object-detection (setup.py): started
  Building wheel for object-detection (setup.py): finished with status 'done'
  Created wheel for object-detection: filename=object_detection-0.1-cp36-none-any.whl size=1368699 sha256=9f843e405d82
  Stored in directory: /tmp/pip-ephem-wheel-cache-ve1thhh3/wheels/94/49/4b/39b051683087a22ef7e80ec52152a27249d1a644ccf
Successfully built object-detection
Installing collected packages: object-detection
Successfully installed object-detection-0.1
```

```
import numpy as np
import os
import six.moves.urllib as urllib
import sys
import tarfile
import tensorflow as tf
import zipfile
```

```
from collections import defaultdict
from io import StringIO
from matplotlib import pyplot as plt
from PIL import Image
from IPython.display import display



from object_detection.utils import ops as utils_ops
from object_detection.utils import label_map_util
from object_detection.utils import visualization_utils as vis_util



# patch tf1 into `utils.ops`
utils_ops.tf = tf.compat.v1

# Patch the location of gfile
tf.gfile = tf.io.gfile


def load_model(model_name):
  base_url = 'http://download.tensorflow.org/models/object_detection/'
  model_file = model_name + '.tar.gz'
  model_dir = tf.keras.utils.get_file(
    fname=model_name,
    origin=base_url + model_file,
    untar=True)

  model_dir = pathlib.Path(model_dir)/"saved_model"

  model = tf.saved_model.load(str(model_dir))

  return model


# List of the strings that is used to add correct label for each box.
PATH_TO_LABELS = 'models/research/object_detection/data/mscoco_label_map.pbtxt'
category_index = label_map_util.create_category_index_from_labelmap(PATH_TO_LABELS, use_display_name=True)
```

```
# If you want to test the code with your images, just add path to the images to the TEST_IMAGE_PATHS.
PATH_TO_TEST_IMAGES_DIR = pathlib.Path('models/research/object_detection/test_images')
TEST_IMAGE_PATHS = sorted(list(PATH_TO_TEST_IMAGES_DIR.glob("*.jpg")))
TEST_IMAGE_PATHS
```

```
[PosixPath('models/research/object_detection/test_images/image1.jpg'),
 PosixPath('models/research/object_detection/test_images/image2.jpg')]
```

```
#Loading an object detection model
model_name = 'ssd_mobilenet_v1_coco_2017_11_17'
detection_model = load_model(model_name)
```

```
INFO:tensorflow:Saver not created because there are no variables in the graph to restore
```

```
print(detection_model.signatures['serving_default'].inputs)
```

```
[<tf.Tensor 'image_tensor:0' shape=(None, None, None, 3) dtype=uint8>]
```

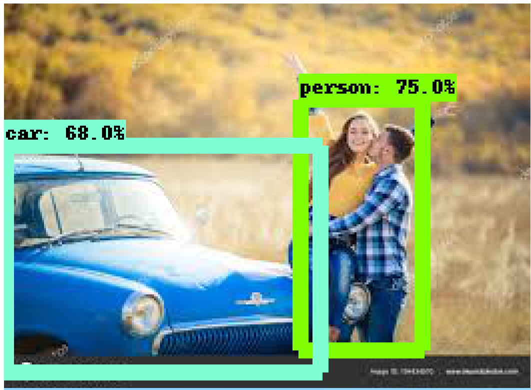
```
detection_model.signatures['serving_default'].output_dtypes
```

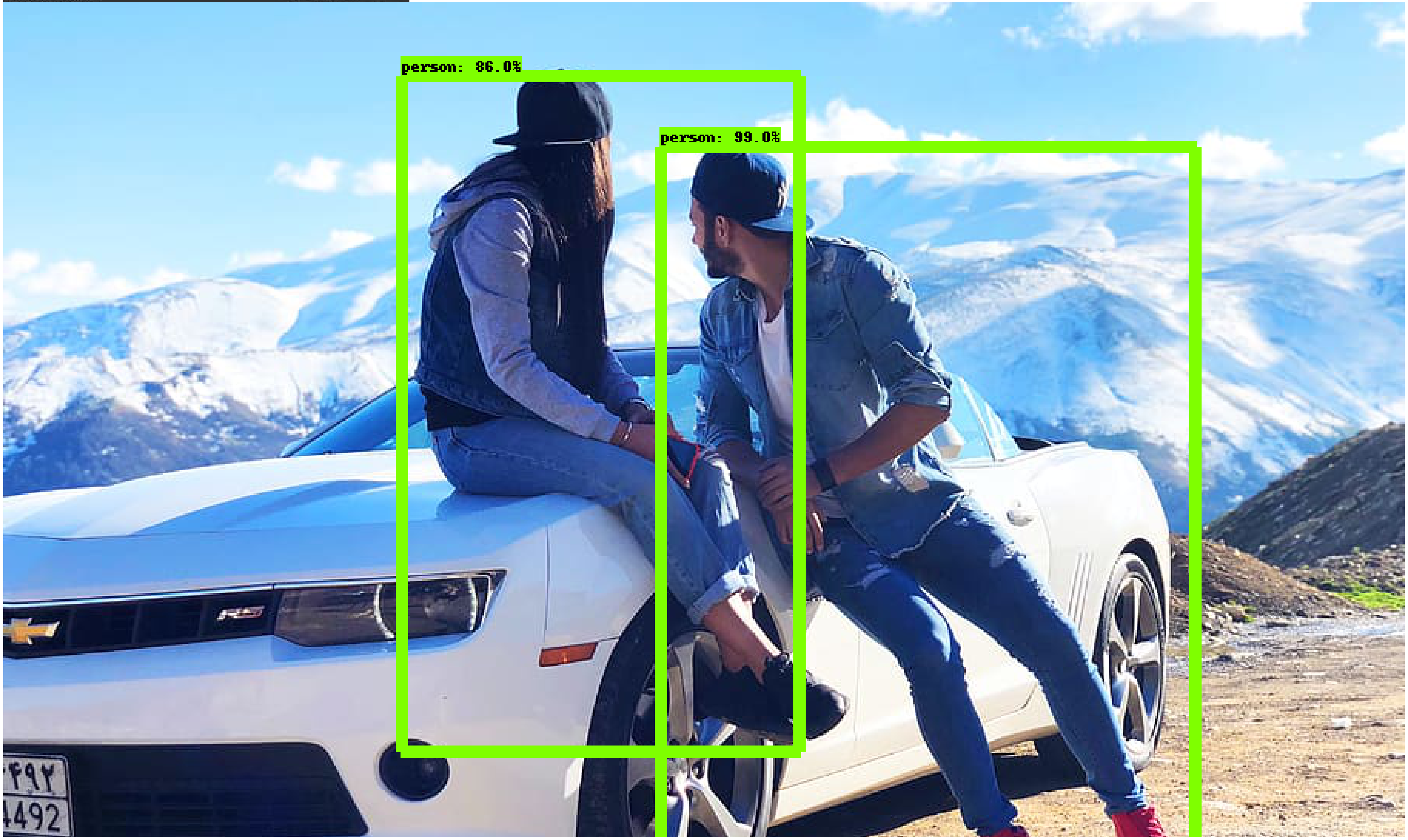
```
{'detection_boxes': tf.float32,
 'detection_classes': tf.float32,
 'detection_scores': tf.float32,
 'num_detections': tf.float32}
```

```
detection_model.signatures['serving_default'].output_shapes
```

```
{'detection_boxes': TensorShape([None, 100, 4]),
 'detection_classes': TensorShape([None, 100]),
 'detection_scores': TensorShape([None, 100]),
 'num_detections': TensorShape([None])}
```

```
def run_inference_for_single_image(model, image):
  image = np.asarray(image)
  # The input needs to be a tensor, convert it using `tf.convert_to_tensor`.
  input_tensor = tf.convert_to_tensor(image)
  # The model expects a batch of images, so add an axis with `tf.newaxis`.
  input_tensor = input_tensor[tf.newaxis,...]

  # Run inference
  model_fn = model.signatures['serving_default']
  output_dict = model_fn(input_tensor)

  # All outputs are batches tensors.
  # Convert to numpy arrays, and take index [0] to remove the batch dimension.
  # We're only interested in the first num_detections.
  num_detections = int(output_dict.pop('num_detections'))
  output_dict = {key:value[0, :num_detections].numpy()
                 for key,value in output_dict.items()}
  output_dict['num_detections'] = num_detections

  # detection_classes should be ints.
  output_dict['detection_classes'] = output_dict['detection_classes'].astype(np.int64)

  # Handle models with masks:
  if 'detection_masks' in output_dict:
    # Reframe the the bbox mask to the image size.
    detection_masks_reframed = utils_ops.reframe_box_masks_to_image_masks(
              output_dict['detection_masks'], output_dict['detection_boxes'],
               image.shape[0], image.shape[1])
    detection_masks_reframed = tf.cast(detection_masks_reframed > 0.5,
                                       tf.uint8)
    output_dict['detection_masks_reframed'] = detection_masks_reframed.numpy()

  return output_dict


def show_inference(model, 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 = np.array(Image.open(image_path))
  # Actual detection.
  output_dict = run_inference_for_single_image(model, image_np)
  # 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_reframed', None),
      use_normalized_coordinates=True,
      line_thickness=8)

  display(Image.fromarray(image_np))


for image_path in TEST_IMAGE_PATHS:
  show_inference(detection_model, image_path)
```


person: 75.0%
car: 68.0%


person: 86.0%
person: 99.0%