SSd_mobileNet with customdata

```
import os
import pathlib
```

```
#Clone the tensorflow model(if it doesn't exists)
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
```

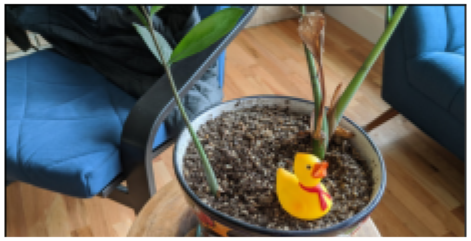
```
# Install the Object Detection API
%%bash
cd models/research/
protoc object_detection/protos/*.proto --python_out=.
cp object_detection/packages/tf2/setup.py .
python -m pip install .
```

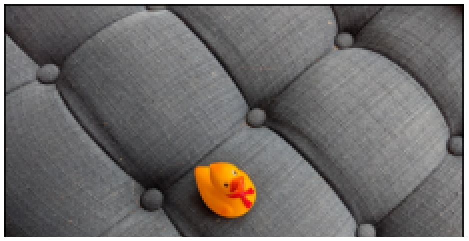
```
Requirement already satisfied: pyasn1>=0.1.7 in /usr/local/lib/python3.6/dist-packages (from oauth2client<5,>=2.0.1-
Requirement already satisfied: pyasn1-modules>=0.0.5 in /usr/local/lib/python3.6/dist-packages (from oauth2client<5,
Requirement already satisfied: rsa>=3.1.4 in /usr/local/lib/python3.6/dist-packages (from oauth2client<5,>=2.0.1->ap
Requirement already satisfied: pbr>=0.11 in /usr/local/lib/python3.6/dist-packages (from mock<3.0.0,>=1.0.1->apache-
Requirement already satisfied: docopt in /usr/local/lib/python3.6/dist-packages (from hdfs<3.0.0,>=2.1.0->apache-bea
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.6/dist-packages (from requests<3.0.0,>=2.

Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.6/dist-packages (from requests<3.0.0,>=2
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.6/dist-packages (from requests<3.0.0,
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.6/dist-packages (from requests<3.0.0,>=2.24.0-
Requirement already satisfied: gast==0.3.3 in /usr/local/lib/python3.6/dist-packages (from tensorflow>=2.3.0->tf-mod
Requirement already satisfied: wrapt>=1.11.1 in /usr/local/lib/python3.6/dist-packages (from tensorflow>=2.3.0->tf-m
Requirement already satisfied: astunparse==1.6.3 in /usr/local/lib/python3.6/dist-packages (from tensorflow>=2.3.0->
Requirement already satisfied: google-pasta>=0.1.8 in /usr/local/lib/python3.6/dist-packages (from tensorflow>=2.3.0
Requirement already satisfied: tensorboard<3,>=2.3.0 in /usr/local/lib/python3.6/dist-packages (from tensorflow>=2.3
Requirement already satisfied: wheel>=0.26 in /usr/local/lib/python3.6/dist-packages (from tensorflow>=2.3.0->tf-mod
Requirement already satisfied: keras-preprocessing<1.2,>=1.1.1 in /usr/local/lib/python3.6/dist-packages (from tenso
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.6/dist-packages (from tensorflow>=2.3.0->
Requirement already satisfied: h5py<2.11.0,>=2.10.0 in /usr/local/lib/python3.6/dist-packages (from tensorflow>=2.3.
Requirement already satisfied: tensorflow-estimator<2.4.0,>=2.3.0 in /usr/local/lib/python3.6/dist-packages (from te
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.6/dist-packages (from tensorflow>=2.3.0->t
Requirement already satisfied: typeguard in /usr/local/lib/python3.6/dist-packages (from tensorflow-addons->tf-model
Requirement already satisfied: tqdm in /usr/local/lib/python3.6/dist-packages (from kaggle>=1.3.9->tf-models-officia
Requirement already satisfied: slugify in /usr/local/lib/python3.6/dist-packages (from kaggle>=1.3.9->tf-models-offi
Requirement already satisfied: python-slugify in /usr/local/lib/python3.6/dist-packages (from kaggle>=1.3.9->tf-mode
Requirement already satisfied: google-cloud-core<2.0dev,>=1.0.3 in /usr/local/lib/python3.6/dist-packages (from goog
Requirement already satisfied: google-resumable-media!=0.4.0,<0.5.0dev,>=0.3.1 in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: google-auth-httplib2>=0.0.3 in /usr/local/lib/python3.6/dist-packages (from google-ap
Requirement already satisfied: google-auth>=1.4.1 in /usr/local/lib/python3.6/dist-packages (from google-api-python-
Requirement already satisfied: uritemplate<4dev,>=3.0.0 in /usr/local/lib/python3.6/dist-packages (from google-api-p
Requirement already satisfied: dm-tree in /usr/local/lib/python3.6/dist-packages (from tensorflow-datasets->tf-model
Requirement already satisfied: importlib-resources; python_version < "3.9" in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: tensorflow-metadata in /usr/local/lib/python3.6/dist-packages (from tensorflow-datase
Requirement already satisfied: attrs>=18.1.0 in /usr/local/lib/python3.6/dist-packages (from tensorflow-datasets->tf
Requirement already satisfied: promise in /usr/local/lib/python3.6/dist-packages (from tensorflow-datasets->tf-model
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /usr/local/lib/python3.6/dist-packages (from tens
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /usr/local/lib/python3.6/dist-packages (from tensorb
Requirement already satisfied: werkzeug>=0.11.15 in /usr/local/lib/python3.6/dist-packages (from tensorboard<3,>=2.3
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.6/dist-packages (from tensorboard<3,>=2.3.0
Requirement already satisfied: text-unidecode>=1.3 in /usr/local/lib/python3.6/dist-packages (from python-slugify->k
Requirement already satisfied: google-api-core<2.0.0dev,>=1.14.0 in /usr/local/lib/python3.6/dist-packages (from goo
Requirement already satisfied: cachetools<5.0,>=2.0.0 in /usr/local/lib/python3.6/dist-packages (from google-auth>=1
Requirement already satisfied: zipp>=0.4; python_version < "3.8" in /usr/local/lib/python3.6/dist-packages (from imp
Requirement already satisfied: googleapis-common-protos<2,>=1.52.0 in /usr/local/lib/python3.6/dist-packages (from t
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.6/dist-packages (from google-auth-
Requirement already satisfied: importlib-metadata; python_version < "3.8" in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.6/dist-packages (from requests-oauthlib>=0.
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=1598976 sha256=a43fc5e125
  Stored in directory: /tmp/pip-ephem-wheel-cache-el5mfgod/wheels/94/49/4b/39b051683087a22ef7e80ec52152a27249d1a644c
Successfully built object-detection
Installing collected packages: object-detection
  Found existing installation: object-detection 0.1
    Uninstalling object-detection-0.1:
      Successfully uninstalled object-detection-0.1
Successfully installed object-detection-0.1
```

```
import matplotlib
import matplotlib.pyplot as plt

import os
import random
import io
import imageio
import glob
import scipy.misc
import numpy as np
from six import BytesIO
from PIL import Image, ImageDraw, ImageFont
from IPython.display import display, Javascript
from IPython.display import Image as IPyImage

import tensorflow as tf

from object_detection.utils import label_map_util
from object_detection.utils import config_util
from object_detection.utils import visualization_utils as viz_utils
from object_detection.utils import colab_utils
from object_detection.builders import model_builder

%matplotlib inline


def load_image_into_numpy_array(path):
  """Load an image from file into a numpy array.

  Puts image into numpy array to feed into tensorflow graph.
  Note that by convention we put it into a numpy array with shape
  (height, width, channels), where channels=3 for RGB.
  Args:
    path: a file path.
  Returns:
    uint8 numpy array with shape (img_height, img_width, 3)
  """
  img_data = tf.io.gfile.GFile(path, 'rb').read()
  image = Image.open(BytesIO(img_data))
```

```
    (im_width, im_height) = image.size
    return np.array(image.getdata()).reshape(
        (im_height, im_width, 3)).astype(np.uint8)

  def plot_detections(image_np,
                      boxes,
                      classes,
                      scores,
                      category_index,
                      figsize=(12, 16),
                      image_name=None):
    """Wrapper function to visualize detections.

    Args:
      image_np: uint8 numpy array with shape (img_height, img_width, 3)
      boxes: a numpy array of shape [N, 4]
      classes: a numpy array of shape [N]. Note that class indices are 1-based,
        and match the keys in the label map.
      scores: a numpy array of shape [N] or None.  If scores=None, then
        this function assumes that the boxes to be plotted are groundtruth
        boxes and plot all boxes as black with no classes or scores.
      category_index: a dict containing category dictionaries (each holding
        category index `id` and category name `name`) keyed by category indices.
      figsize: size for the figure.
      image_name: a name for the image file.
    """
    image_np_with_annotations = image_np.copy()
    viz_utils.visualize_boxes_and_labels_on_image_array(
        image_np_with_annotations,
        boxes,
        classes,
        scores,
        category_index,
        use_normalized_coordinates=True,
        min_score_thresh=0.8)
    if image_name:
      plt.imsave(image_name, image_np_with_annotations)
    else:
      plt.imshow(image_np_with_annotations)
```

```
# Loading images and visualizing it
train_image_dir = 'models/research/object_detection/test_images/ducky/train/'
train_images_np = []
for i in range(1, 6):
  image_path = os.path.join(train_image_dir, 'robertducky' + str(i) + '.jpg')
  train_images_np.append(load_image_into_numpy_array(image_path))

plt.rcParams['axes.grid'] = False
plt.rcParams['xtick.labelsize'] = False
plt.rcParams['ytick.labelsize'] = False
plt.rcParams['xtick.top'] = False
plt.rcParams['xtick.bottom'] = False
plt.rcParams['ytick.left'] = False
plt.rcParams['ytick.right'] = False
plt.rcParams['figure.figsize'] = [14, 7]

for idx, train_image_np in enumerate(train_images_np):
  plt.subplot(2, 3, idx+1)
  plt.imshow(train_image_np)
plt.show()
```





```
# Manually adding bounding box for the images

gt_boxes = [
            np.array([[0.436, 0.591, 0.629, 0.712]], dtype=np.float32),
            np.array([[0.539, 0.583, 0.73, 0.71]], dtype=np.float32),
            np.array([[0.464, 0.414, 0.626, 0.548]], dtype=np.float32),
            np.array([[0.313, 0.308, 0.648, 0.526]], dtype=np.float32),
            np.array([[0.256, 0.444, 0.484, 0.629]], dtype=np.float32)
]
```

```
# Training the data

# By convention, our non-background classes start counting at 1.  Given
# that we will be predicting just one class, we will therefore assign it a
# `class id` of 1.
duck_class_id = 1
num_classes = 1

category_index = {duck_class_id: {'id': duck_class_id, 'name': 'rubber_ducky'}}

# Convert class labels to one-hot; convert everything to tensors.
# The `label_id_offset` here shifts all classes by a certain number of indices;
# we do this here so that the model receives one-hot labels where non-background
# classes start counting at the zeroth index.  This is ordinarily just handled
# automatically in our training binaries, but we need to reproduce it here.
label_id_offset = 1
train_image_tensors = []
gt_classes_one_hot_tensors = []
gt_box_tensors = []
for (train_image_np, gt_box_np) in zip(
    train_images_np, gt_boxes):
```

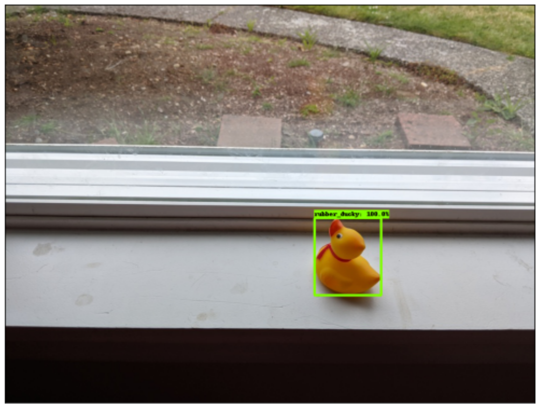
```
    train_image_tensors.append(tf.expand_dims(tf.convert_to_tensor(
        train_image_np, dtype=tf.float32), axis=0))
    gt_box_tensors.append(tf.convert_to_tensor(gt_box_np, dtype=tf.float32))
    zero_indexed_groundtruth_classes = tf.convert_to_tensor(
        np.ones(shape=[gt_box_np.shape[0]], dtype=np.int32) - label_id_offset)
    gt_classes_one_hot_tensors.append(tf.one_hot(
        zero_indexed_groundtruth_classes, num_classes))
  print('Done prepping data.')
```

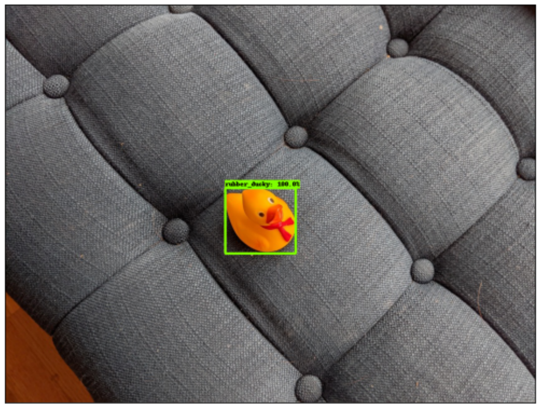
```
Done prepping data.
```

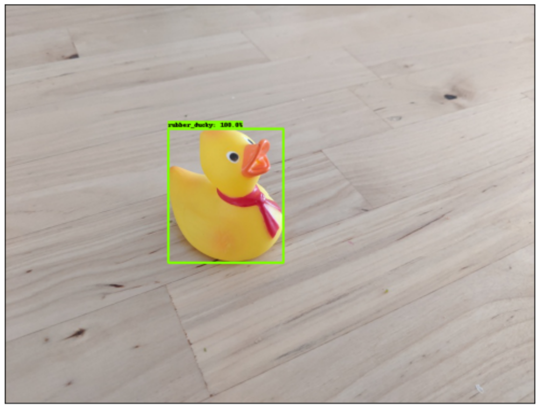
Visualizing the data

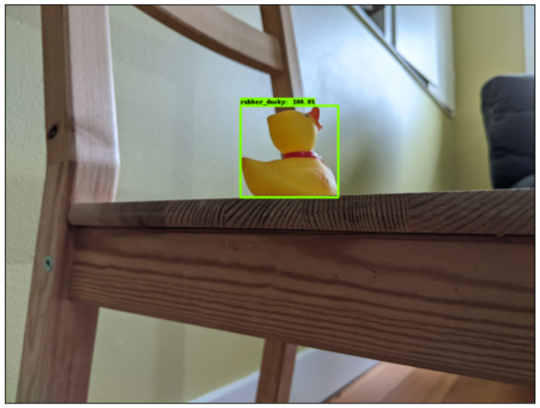
```
dummy_scores = np.array([1.0], dtype=np.float32)  # give boxes a score of 100%

plt.figure(figsize=(30, 15))
for idx in range(5):
  plt.subplot(2, 3, idx+1)
  plot_detections(
      train_images_np[idx],
      gt_boxes[idx],
      np.ones(shape=[gt_boxes[idx].shape[0]], dtype=np.int32),dummy_scores, category_index)
plt.show()
```

rubber_ducky: 100.0%


rubber_ducky: 100.0%


rubber_ducky: 100.0%


rubber_ducky: 100.0%


rubber_ducky: 100.0%