First they should follow the below link till the end and reproduce everything:
1. https://gilberttanner.com/blog/yolo-object-detection-with-keras-yolo3

Second they should follow the below link and reproduce everything:
2. https://pjreddie.com/darknet/yolo/

Third, The following link just for yet another reference:
https://github.com/experiencor/keras-yolo3

Also, in 1 as they had used old version of keras, they would face some difficulty:

Following are the changes they would require as and when they face difficulty:

Lines to be changed in yolo_video.py:

parser.add_argument((remove)'--model', type=str,

(add) '-m','--model_path', type=str,

help='path to model weight file, default ' + YOLO.get_defaults("model_path"))

parser.add_argument((remove)'--anchors', type=str,

(add) '-a','--anchors_path', type=str,

help='path to anchor definitions, default ' + YOLO.get_defaults("anchors_path"))

parser.add_argument((remove)'--classes', type=str,

(add) '-c','--classes_path', type=str,

```
                help='path to class definitions, default ' + YOLO.get_defaults("classes_path"))
```

Also:

Add 2 lines at the top:

import tensorflow.compat.v1 as tf1

tf1.disable_v2_behavior()

Also:

Add break in the line next to line number 18 (r_image.show())

Also:

import cv2 and use cv2.imwrite to write annotated image as it is not being displayed in colab.

Changes in yolo.py:

Comment line: keras.utils import multi_gpu_model

Add line: import tensorflow.python.keras.backend as K

Change Line:

From:

video_FourCC = int(vid.get(cv2.CAP_PROP_FOURCC))

to

video_FourCC = cv2.VideoWriter_fourcc(*"mp4v")

Comment 2 lines:

cv2.namedWindow("result", cv2.WINDOW_NORMAL)

and

cv2.imshow("result", result)

After Line return_value, frame = vid.read()

Add:

if not return_value: break

Changes in yolo3/model.py:

Change line number 140/141 from:

box_xy = (K.sigmoid(feats[..., :2]) + grid) / K.cast(grid_shape[::-1], K.dtype(feats))

box_wh = K.exp(feats[..., 2:4]) * anchors_tensor / K.cast(input_shape[::-1], K.dtype(feats))

to:

box_xy = (K.sigmoid(feats[..., :2]) + grid) / K.cast(grid_shape[...,::-1], K.dtype(feats))

box_wh = K.exp(feats[..., 2:4]) * anchors_tensor / K.cast(input_shape[...,::-1], K.dtype(feats))

For Custom Microcontroller Dataset:

First training and test targets are generated using custom_voc_to_yolo.ipynb Row format: image_file_path box1 box2 ... boxN; Box format: x_min,y_min,x_max,y_max,class_id (no space). (They need to write this ipynb file)

Here is an example:

path_to_img1.jpg 50,100,150,200,0 30,50,200,120,3

path_to_img2.jpg 120,300,250,600,2 ...

In train.py following changes are to be made:

set annotation_path and classes_path

batch_size changed to 4 at two places

You can also set number of epochs

In model.py following changes are to be made:

tf.while_loop to be used in place of K.control_flow_ops.while_loop in line 394

In yolov3.cfg following changes are to be made:

In 3 [yolo] layers, change classes=4 from classes=80 Before each of the 3 [yolo] layers, change filters=27 (1 pc + 4 bb + 4 class)*3 anchors