Lab 6: Object Detection with Deep NN

(10% of final score, due by ~~Apr. 8~~ Apr. 9, 11:59 PM)

Lab 6 will be implemented on Google CoLab. Please keep the code file output when saving the notebook and share the code with: abbas.salehitangrizi@gmail.com. Please add comments if needed and follow the PEP-8 Style Guide for Python Code. [https://www.python.org/dev/peps/pep-0008/]

1. Convolutional Neural Network (CNN): https://www.tensorflow.org/tutorials/images/cnn

   Read the example file < cnn.ipynb> , save it to your G-drive, and run it in Google CoLab. Answer the following questions:

   a) How is the sequential model defined?
   b) Read the documentation of tf.keras.layers.Conv2D. https://www.tensorflow.org/api_docs/python/tf/keras/layers/Conv2D
      If the inputs are 50×50 RGB images, *y*= tf.keras.layers.Conv2D( ), what would the y.shape be with: batch size = 4, strides = 2 , and padding="valid"
   c) Before adding two dense layers, how is the number of total parameters 56,320 calculated?
   d) After the sequential model, why are two dense layers added?
   e) After the two dense layers, the number of total parameters is 122,570. How is it calculated?
   f) This CNN achieved a test accuracy of about 70%. Is it a good number? How to improve it?

2. Object Detection Models: https://www.tensorflow.org/hub/tutorials/tf2_object_detection

   Read the example file <Object Detection Inference on TF 2 and TF Hub.ipynb>, save it to your G-drive, and run it in Google Colab.

   a) Compare models: “Mask R-CNN Inception ResNet V2 1024×1024” and “EfficientDet D0 521×512.” Evaluation metric: inference time. Using the same test image. Describe your observation and explain it.
   b) Do some research on the models “Mask R-CNN Inception ResNet V2 1024×1024” and “EfficientDet D7 1536×1536” and briefly (one paragraph for each) describe the mechanism of the models. Why is EfficientDet very efficient? Why does Mask R-CNN allow instance segmentation? Here is the link to the original publications: https://openaccess.thecvf.com/content_CVPR_2020/papers/Tan_EfficientDet_Scalable_and_Efficient_Object_Detection_CVPR_2020_paper.pdf
      https://openaccess.thecvf.com/content_ICCV_2017/papers/He_Mask_R-CNN_ICCV_2017_paper.pdf

3. Transfer learning:
   https://colab.research.google.com/github/tensorflow/hub/blob/master/examples/colab/tf2_image_retraining.ipynb

   Read the example file <TF Hub for TF2: Retraining an image classifier.ipynb>, save it to your G-drive, and run it in Google Colab.
   a) model.summary() outputs some parameter are trainable while others are non-trainable.Why? Check the box "do_fine_tuning" and its value will be changed to True. Run the model again. The outputs of model.summary() will change and the number of trainable parameters will significantly increase. Why?
   b) The default batch_size is 16. The parameters steps_per_epoch and validation_steps vary with batch_size. Adjust batch_size to 8, 4, and 2, observe inference time, loss, and accuracy, and describe the changes and explain why?
   c) Try out the model on your input images with and without flowers (not from the validation data). Are the predictions correct?