ECE 310 - Machine Learning Engineering Practicum - Spring 2023-2024
Assignment 3: Transfer Learning for Image Classification
Prof. John MacLaren Walsh, Ph.D.

1 Aim of this Assignment

The goal of the assignment is to practice rapidly prototyping and testing transfer learning for the types of image classification problems you may encounter professionally.

2 Dataset and Your Objectives

You work for a company that provides software and gear for natuarlists and bird enthusiasts. You are to help with two development projects - a flowers identification app, and a birdfeeder monitoring system.

The flowers identification app classifies pictures of flowers among the 102 types in the oxford flowers dataset. The birdfeeder monitoring system detects the presence of birds at the feeder, as well as detects when squirrels or chipmunks are trying to eat from it. The feeders the company builds have doors that can be remotely controlled by the integrated camera system to open or close via WiFi.

Accordingly, your initial assignment working for the company is to build two vision architectures to classify still images. The first recognizes the 102 different types of flowers in the oxford flowers dataset. The second classifies images into three categories: 1 - containing a bird and no squirrel/chipmunk, 2 - containing a squirrel/chipmuck, or 3 - containing a empty bird-feeder/bird-house/tree. To help train your birdfeeder monitoring architecture you have been given a tfrecords dataset consisting of two collections of images of 3 types (containing squirrels/chipmunks, containing birds without squirrels/chipmunks, containing trees/birdfeeders/birdhouses without birds/squirrels/chipmunks). Please see the template code in §**??** for how to load this tfrecord file into the tensorflow data API. You will train your flowers model using the training and validation portions (only) of the oxford flowers 102 dataset.

3 Tasks You Must Complete

You will utilize transfer learning to build, train, and save, two deep neural network models in keras on TensorFlow.

- **Flowers Model**: Classifies floral images into the 102 categories in the Oxford flowers 102 dataset.
- **Birds vs Squirrels Classifier**: This model will be designed to decide between three outcomes:
  - a bird and no squirrel/chipmunk is present in the image
  - a squirrel is present in the image
  - the image is of a tree/birdfeeder/birdhouse that has no squirrel or bird nearby.

4 What You Must Submit

Your submission should be a gzipped or zipped archive titled **abc123-lab3.tgz** or **abc123-lab3.zip** where abc123 is replaced with your Drexel userid. The archive must contain each of the following files

1. preprocessDefinition.py: defines the function preprocess that can be applied in a map command on the dataset to make heterogenous images into inputs compatable with your two models.
2. buildAndTrainFlowersModel.py: loads the oxford flowers 102 dataset within tensorflow, defines a Keras model for classifing floral images among these 102 categores, trains this model and saves this model. Note your training process may only access the training and validation sets of the original dataset.
3. buildAndTrainBirdsVsSquirrels.py: loads relevant data, preprocesses it, defines model for (bird-no-squirrel/chipmunk, squirrel/chipmunk, feeder/tree/house-w/o-bird/squirrel/chipmunk) three-way classification, trains the model, saves the model.
4. flowersModel.tgz or flowersModel.zip a compressed archive of the saved model directory created by the model.save command applied to your model to classify floral images.
5. birdsVsSquirrelsModel.tgz or birdsVsSquirrelsModel.zip a compressed archive of the saved model directory created by the model.save command applied to your model to classify images into the three categories (bird-no-squirrel/chipmunk, squirrel/chipmunk, feeder/tree-w/o-squirrel-or-bird).

5 How Your Assignment Will be Graded

Your grade will consist of the following assessments which will be totaled to create a final score

1. Compliance - did your submission have all of the necessary files, and were they named appropriately.
2. Preprocess Code - When placed into a map command applied to the loaded dataset, does your myPreprocess function produce outputs compatable with your models?
3. Model Training Code assessments: in the code buildAndTrainbirdsVsSquirrelsModel.py and buildAndTrainFlowersModel.py, do you complete at least each of the following
   (a) loads the data, preprocesses it
   (b) load a pre-trained model to transfer from
   (c) modify its top in a manner that includes at the end an appropriate output layer for the task and other hidden layers as you wish.
   (d) freeze the pre-trained weights
   (e) compile the model with an appropriate loss function, optimizer, and metrics
   (f) fit the model
   (g) save the final trained model
4. Birds versus Squirrels Model Performance - How do the top1, top2, and top 5 accuracies of your model saved in flowersModel.tgz or the model trained by your buildAndTrainFlowersModel.py compare with the accuracy of that of your peers? See Section **??** for example evaluation code.
5. Birds versus Squirrels Model Performance - How does the accuracy of your model saved in birdsVsSquirrelsModel.tgz or the model trained by your buildAndTrainBirdsVsSquirrelsModel.py compare with that of your peers?

5.1 Template Code for Loading Birds vs. Squirrels Data

```
import tensorflow as tf
from preprocessDefinition import preprocess

raw_dataset=tf.data.TFRecordDataset(['birds-vs-squirrels-train.tfrecords'])

feature_description={'image':tf.io.FixedLenFeature([],tf.string),
                     'label':tf.io.FixedLenFeature([],tf.int64)}

def parse_examples(serialized_examples):
    examples=tf.io.parse_example(serialized_examples,feature_description)
    targets=examples.pop('label')
    images=tf.image.resize_with_pad(tf.cast(tf.io.decode_jpeg(
        examples['image'],channels=3),tf.float32),299,299)
    return images, targets

#you can edit batch size and num_parallel calls below based on your architecture
dataset=raw_dataset.map(parse_examples,num_parallel_calls=16).batch(128)

# run through your provided preprocess method
dataset=dataset.map(preprocess,num_parallel_calls=16).cache()
```

5.2 Template Code for Flowers Model Performance Measurement

```
import tensorflow as tf
from preprocessDefinition import preprocess
import tensorflow_datasets as tfds
from tensorflow import keras
from preprocessDefinition import preprocess

#load your flower model
model=tf.keras.models.load_model('flowersModel')

#load the dataset and preprocess it
evalset,info = tfds.load(name='oxford_flowers102', split='test',as_supervised=True,with_info=True)
evalPipe=evalset.map(preprocess,num_parallel_calls=16).batch(32).prefetch(1)

#ensure the metrics included are the ones we wish to evaluate
top2err=tf.keras.metrics.SparseTopKCategoricalAccuracy(k=2,name='top2')
top5err=tf.keras.metrics.SparseTopKCategoricalAccuracy(k=5,name='top5')

model.compile(loss='sparse_categorical_crossentropy',optimizer=opt,
        metrics=['accuracy',top2err,top5err])

#returns the loss and metrics evaluated on the dataset
resp=model.evaluate(dataset)
```