SYSC4906 Introduction to Machine Learning Fall 2019

Assignment 3 - **Name That Building!**

Preamble

Visitors to Carleton University often find it difficult to navigate the campus and identify buildings from their exteriors. This is particularly true for visitors with vision-related disabilities. In this project, we aim to develop a robust building classification system that will correctly name any of 16 buildings on campus, when given a square daylight photo of the building from an arbitrary angle. The class will work together to crowdsource a collection of photos of each building. Working in teams of 2, you will leverage these photos to develop your solution in a Jupyter notebook with optional GPU acceleration. You will also need to estimate your system's performance on future test images, as detailed below.

Buildings

Most of the buildings on campus are described at: https://carleton.ca/campus/buildings/. As agreed in class, we will include the following 16 buildings in our system:

Name	**Label**	**Name**	**Label**
Architecture	AA	Minto Centre	MC
Canal Building	CB	Mackenzie Building	ME
CTTC	CT	MacOdrum Library	ML
Dunton Tower	DT	Patterson Hall	PA
Fieldhouse	FH	Richcraft Building	RB
Herzberg Laboratories	HP	Robertson Hall	RO
Health Sciences Building	HS	Southam Hall	SA
Loeb Building	LB	Tory Building	TB

Detailed Instructions

Step 0: Form a Group

The assignment will be completed in groups of 2. Please create or join a group using the activity on the course CULearn page.

Step 1: Data Collection

<u>Each student</u> is to collect 5 photographs of each of the 16 buildings. Details:

- Each photo should have a square aspect ratio and be of at least 500x500 pixels
- Each photo should be taken during the day
- Each of the 5 photos of each building should represent a different view
    - *(don't just stand in one spot and click 5 photos)*
- Each photo should contain a single building, as much as possible. The building in question should be centered in the image

Upload your images to GitHub at
https://github.com/jrgreen7/SYSC4906/tree/master/Assignments/Assignment3/Images

- There are 16 sub-directories, each named with a building code. Please upload images into the correct location.
- **Name each image as XYZ_n.jpg**, where *XYZ* are last 3 digits of your student number and *n* ranges from 1-5 (1st to 5th photo of each building)
- *Note that some students expressed concerns about the upload speed when using GitHub as a repository for images. If you install GitHub Desktop on a PC, you can clone the Assignment3 repo, add your images locally, then push your new images back to GitHub in a single action. On Sunday, you can pull down everyone's images to your local drive with a single sync, and continue without GitHub.*

All images must be uploaded by 9pm Sat Nov 23.

Step 2: Train Your Classifier

- You are free to use any machine learning methods in the creation of your image classifier.
- Constraints:
    - Your solution must accept a square image of size at least 500x500 pixels and return a building label from the table above.
    - Your method should not use image meta-data (e.g. GPS coordinates, etc).
    - Your method should classify new images in less than 1 minute per test image, when run on Google Colab using GPU acceleration.

- Your notebook can load external files from URLs
    - e.g. your pre-trained model could be loaded from a file on GitHub or another web-accessible location.
        - Note that your own Google Drive will not be accessible to us when evaluating your method, so avoid using it for this purpose.

Step 3: Make Your Prediction!

Using the images available to you, you must predict how well your system will perform on a collection of new test images. Specifically, you must:

1) Predict the overall accuracy (0.0-1.0) that your system will obtain on a class-balanced collection of new test images
2) Predict which building you will have the lowest recall for. You must choose a single building.
3) Predict which building you will have the highest precision for. Here again, you must choose a single building.

These predictions should be encoded in two ways: in the mandatory 500 word text report at the top of your notebook (see below) and in the three access functions `worstRecall()`, `bestPrecision()`, and `estimatedAccuracy()`. See below for more details.

Step 4: Submit your Notebook

- Your submitted Jupyter notebook should include all of the code you use to train and test your solution.
- Your notebook can have multiple functions, but <u>must</u> implement the following 5 functions:
    - `prepareModel()`
        - This function should prepare your model for multiple invocations of `classifyImage(fname)`. For example, this function could be used to load a pre-trained model from a URL, where that model is then used by `classifyImage(fname)`. You should use **global** variables for any variables initialized by this function.
        - Runtime of this method is limited to 5 minutes, so please don't retrain your network here. All training should be captured in a pre-trained model to be loaded by this method.
    - `label = classifyImage(fname)`
        - Accepts a filename (e.g. 'test/ME/testImage1.jpg') of a square JPG image with size at least 500x500 pixels.
            - Your function must take care of all required preprocessing (rotating, resizing, normalizing, etc).
        - Returns a 2-character label corresponding to the predicted building (see table of labels above)

        - Any variables initialized by `prepareModel()` should be declared as **global** within this function if you want to access them (e.g. a pre-trained model)
    - `label = worstRecall()`
        - Returns the label of a building that you expect will have to lowest recall, when tested on new images
    - `label = bestPrecision()`
        - Returns the label of a single building that you expect will have the highest precision, when tested on new images.
    - `acc_score = estimatedAccuracy()`
        - Returns the accuracy (between [0.0,1.0]) that you expect to achieve across all test images, assuming that each building is equally represented
- Your notebook must also **begin** with a text cell containing a description of your solution. In your discussion, include links to any resources that you used in developing your solution. Use proper MarkDown syntax to format your discussion.This description should be approximately 500 words in length and cover the following:
    - 1) Which machine learning approach did you use?
    - 2) How did you split your data between training and testing
        - (e.g. *hold-out test, cross-validation, repeated bootstrap samples, etc*)
    - 3) How did you train your classifier?
        - If you used transfer learning, describe how you did so
    - 4) How did you estimate your future performance (worst recall, best precision, overall accuracy)?
        - Also state your predictions (building with worst recall, building with best precision, overall accuracy)
    - 5) Discuss the performance of your model. Which buildings did it do the best/worst on and why? What are the strengths and limitations of your method.
    - 6) What would you have done differently if you had more time?

You must submit your final notebook to CULearn by 11:55pm on Tuesday 3 December.

Step5: Your Method is Tested

- We will collect our own images of each of the 16 buildings. We will have an equal number of images from each building. They will all be square and of at least 500x500 pixels in size and saved in JPG format.
- To test your notebook:
    - We will load our images and your notebook into Google Colab
    - We will invoke your `prepareModel()`function once. It must complete within 5 minutes.
    - We will then loop through all of our test images, invoking `classifyImage(fname)`on each image file. Each file cannot take longer than 1 minute to classify.

    - We will invoke `worstRecall()`, `bestPrecision()`, and `estimatedAccuracy()` to obtain your predicted performance.

You will be evaluated on:

- Did you upload 5 images of the 16 buildings by the deadline?
- Do your 5 functions work? Were we able to apply your solution to each test image. If not, you will score very poorly on the assignment.
- You will be marked on your textual description of your solution, including your answers to the 6 questions above. You will be marked on both content and style/grammar.
- You will be marked on your overall achievable accuracy.
- You will be marked on how well your performance predictions (best, worst, average) match your actual performance on the test images.

Step 6: Gloat

- All team performances on the test images will be revealed in-class on Friday 6 Dec.
- There will be prizes for the top-performing team(s), handed out in class.
    - Those teams will also be asked to discuss their method for a few minutes, explaining why they feel their solution was particularly effective.
- **Good luck!!**

Resources for this problem:

- See the GitHub repo at https://github.com/jrgreen7/SYSC4906/tree/master/Assignments/Assignment3 for:
    - These instructions in PDF
    - As of the morning of 24 November, you can collect all building images from the `Images` sub-directory
    - A template notebook for your solution
- Description of all pre-trained deep learning models available in Keras: https://keras.io/applications/
    - Note that you can use any machine learning approach you like, so long as it requires only a square image as input
- Tutorial on transfer learning and data augmentation using InceptionV3 model:
    - https://medium.com/abraia/first-steps-with-transfer-learning-for-custom-image-classification-with-keras-b941601fcad5
- Tutorial on transfer learning: https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751
- Tutorial on image data augmentation: https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/