## HU Extension Assignment 06 E89 Deep Learning

### Handed out: 10/10/2020 Due by 11:59 AM EST on Saturday, 10/17/2020

You are provided with 2 Jupyter notebook created by Francois Chollet: 5.2-using-convnets-with-small-dateset.ipynb and 5.3-using-a-pretrained-convnet.ipynb. Please save an unmodified copy of each of those Jupyter notebooks so that you can compare his (original) results with your own.

Before you start running your Jupyter notebooks, please install Python packages: pillow and h5py.

**Problem 1.** Examine the CNN for the basic binary classification on Kaggle.com dog\_vs\_cats dataset, as provided in the Jupyter Notebook 5.2-using-convnets-with-small-dateset.ipynb. That CNN has 4 convolutional layers, 4 max\_pooling layers, 1 flatten layer and 2 dense layers. Keep original cells with code and results unmodofoed so that you could use them for comparison. Add new cells for modified definition and training of your model. Add an L1 regularizer on next to the last Dense layer. That is the Dense layer with (512,1) output tensor. As the regularization lambda, use the value of 0.0001. Run your model for 15 epochs and compare it with the original result on the model without regularization. Present the plots of training and validation accuracies and let us know whether regularization alleviate the overfitting and to what extent. Save your model as an H5 file. Keep H5 file for future reference. Do not submit H5 file as part of your solution.

**(20%)**

**Problem 2.** Add new cells to previous notebook. This time, change the regularizer to L1L2. Compare the results of models with no regularization and L1 and L1L2 regularization.

**(10%)**

**Problem 3.** Add new cells to your notebook and extend your model with L1 regularization with three callbacks: EarlyStopping, ModelCheckpoint and TensorBoard.Apply all three to your model.fit() invocation as a single callback list. Capture accuracy and loss plots for the model. Capture TensorBoard images of those same scalars. If you know how add captured TensorBoard images to your notebook. If you do not, submit them as one JPG file.

(**10%)**

**Problem 4.** Demonstrate image augmentation for an image of your choice. Produce one modified image for every one of these options separately: rotation\_range, width\_shift, shear\_range, zoom\_range, vertical\_flip, and horizontal\_fit. Reference notebooks generate images which have random transformations along all of those option axes. You are asked to perform one transformation at a time.

**(10%)**

**Problem 5.** In the first portion of Jupyter notebook 5.3-using-a-pretrained-convnet.ipynb the author applied Transfer Learning or Feature Extraction technique by collecting the output of the convolutional base and then training simple Dense classifier (cell 6). In the class a question was raised whether we could add a convolutional layer at the bottom of the dense classifier and whether that would improve the accuracy of the model. Please keep the original code and results unmodified (cells 6 and 7) for comparison with your results. Add new cells to the notebook with an extended trainable layer which besides the dense top has at least one convolutional layer. Report on the number of trainable parameters in your model.

**(25%)**

**Problem 6**. In the last portion of Jupyter notebook 5.3-using-a-pretrained-convnet.ipynb we have results for a fine-tuned VGG16 network. Chollet fine-tuned the last 3 conv layers: block5\_conv1, block5\_conv2 and block5\_conv3. Examine whether fine-tuning layerblock4\_conv3 only and freezing layers in block 5: block5\_conv1, block5\_conv2 and block5\_conv3 will result in significant change for better or worse in the training accuracy. Report the number of trainable parameters when Chollet was fine tuning the last 3 convolutional layers in block5 vs. when you are tuning only one layer block5\_conv3. Please leave the original code and results in the notebook unmodified so that you can easily make the comparison. You might have to search through Keras documentation on how to freeze and unfreeze particular convolutional layers.

**(25%)**

Please keep your saved models for your future use. Please do not upload them as parts of your solution.

Please add your name to the top of your notebooks.

We expect you to submit two notebooks with their HTML or PDF images.

Problems 4 and 5 require GPU processing. Unless you have a GPU card yourself, do those problems in Google Colab or AWS.

If you are using AWS, please note thatAWS charges per second of the use of EC2 instances.

This is very important. You can perform brief experiments and then stop your instance while analyzing results or contemplating code change. If and when selecting an AWS instance, try offering 1/3 or ½ normal price by choosing a spot instance. Spot instances are not always available.

Run compilation of all your code and training on one or two epochs on your local machine and only when you are sure that your code works, transfer the code to the AWS instance of your choice. Pay attention to the amount of time your instance is running. Once you are done with a particular test or a training run, stop any AWS instance used. Stopped instances are not very expensive.