The University of Texas at Austin
Cognitive Computing — Spring 2019

Homework One

Caramanis/Dimakis Due: Wednesday, February 13, 2019.

The goal of this problem set is to get us up and running with some modern tools for playing with Neural Networks. *This problem set will require access to a GPU.* Note that the recommended (and easiest) way to run this is through the cloud service `paperspace.com`. You should be able to complete this with just a few hours of compute. Paperspace charges $0.78 per hour for a very strong machine. But there are (less easy) ways to complete this assignment which can leverage free credits on colab or some other service of your choice.

Note that there is nothing to hand in for Questions 1-3, but there is for Question 4-7.

Programming Questions

1. **Paperspace**

   Create an account on paperspace or a cloud service of your choice. You should make sure that you are able to install all the libraries and tools needed for Fast.ai. If you choose to do this on `paperspace.com`, then:

   (a) Create an account on paperspace.

   (b) Log in.

   (c) Go to Gradient in the toolbar in the left.

   (d) Create Notebook.

   (e) Paperspace Fast.AI 1.0 (V3).

   (f) Choose the P5000 machine ($0.78/hour).

   (g) Launch, and go to: course-v3 $\longrightarrow$ nbs $\longrightarrow$ dl1 and open up the first lesson: lesson1-pets.ipynb.

2. **Fast.ai**

   Fast.ai has created a huge library of tools that make setting up and training a neural network very easy, especially through the use of *transfer learning*. The notebook you opened in the previous exercise corresponds to the first lesson under "Practical Deep Learning for Coders." Watch this lesson, and follow along on the notebook.

3. Make sure you understand the key elements of using Fast.ai, including understanding how to access the help documentation. Specifically:

   (a) Figure out: ImageDataBunch: this is the main data structure that is used.

   (b) Figure out: create_cnn. This is the way you will create a "learner" that you will then train for some number of epochs. The key here is how easy it is to download a pre-set architecture (e.g., the notebook starts with resnet34, but has various others as well).

4. Explain what is a residual network, and the basic motivation for using it. Also explain what are the main elements of resnet34 and resnet50. How many layers, how many neurons total, how many weights; and then anything else you want to say.

5. Transfer learning using Fast.ai and create_cnn: Please explain how *pretrained* resnet34 is modified to get the network that the notebook ultimately trains (i.e., explain what are the last layers that are added).

6. Download a NOT pre-trained resnet34, and then by playing with the number of epochs and learning rates (possibly different learning rates across layers), see how low you can get the error. Can you get below 20%?

7. And for the main part of this HW: download (and label) your own data set of your choie, create a classification problem, and then use the main tools/ideas of this notebook to build a classifier. It does not need to be a multi-label classifier.

   For getting data, you may want to refer to the discussion here, for various tools that could be useful: `https://forums.fast.ai/t/tips-for-building-large-image-datasets/26688/36`.