COMP 543 Lab: Deep Learning with TensorFlow

1 Description

In this lab, you will be using Google’s open-source TensorFlow machine learning tool to implement a deep learning architectures that will be used to classify sequences of raw text (This time, no pre-processing using dictionaries and words. We will be operating on raw characters!). Since deep learning is computationally expensive, it is strongly recommended that you use one of Amazon’s deep learning machines to do this assignment (see below). The small learning problems we’ll consider will take about 10-20 minutes max using one of these machines, but might take 10 times as long (or more) using a laptop. Plus, TensorFlow can be a bit of a pain to install on your laptop, so using Amazon is just plain easier. The only situation in which you might consider using your own hardware is if you’ve got a tricked out laptop/desktop with a beefy GPU for game playing that TensorFlow can make use of.

2 The Task: Running RNN Learning Using TensorFlow

For this task, you’ll make a slight modification to a Python TensorFlow code provided, and then you’ll run it on EC2 (this is Amazon’s computer rental service; you already have experience running EC2, although previously you used EC2 via Amazon’s EMR service; you didn’t start up machines using EC2 directly).

To get started, log on to Amazon AWS, then click on “EC2”, and “Launch Instance”. You will be asked for a machine instance to start up (this will govern the software on the machine you run). Scroll down to “Deep Learning AMI (Ubuntu) Version [X] - ami-[id]” and click “Select”. Make sure you select a Deep Learning AMI, not a Deep Learning Base AMI. This machine instance (not surprisingly, given the name) has a number of deep learning tools installed on it. Next you need to choose the machine type you will rent. You will want to choose a machine with a GPU, which will make your deep learning codes much faster. Choose “p2.xlarge“, then “Next: Configure Instance Details“, please make sure that you set IAM role as “EMR_EC2_DefaultRole“, then “Review and Launch”. You want to make sure that you can SSH into your machine, so choose “Edit Security Groups” then click “Add Rule” to make sure to allow SSH access. Once you have done that, click “Review and Launch” and then “Launch”. You can find your machine by going to the EC2 Dashboard and then clicking on “Running Instances”.

As usual, when you are done with your machine, **MAKE SURE TO SHUT IT DOWN!!**

Once you have a machine up and running, SSH into your machine (just like for EMR) then load up three of the data sets from the first Lab:

```
aws s3 cp s3://risamyersbucket/text/Holmes.txt ./Holmes.txt
aws s3 cp s3://risamyersbucket/text/war.txt ./war.txt
aws s3 cp s3://risamyersbucket/text/william.txt ./william.txt
```

Our goal is to implement a deep learner that is able to accept lines of text from each of those three files and classify the line correctly (that is, accurately determine which file the line came from). To do this, we will fire up Python since TensorFlow has a Python API. Start by typing:

```
   ubuntu@ip-172-16-0-163:  source activate tensorflow_p36
```

This will give you a virtual environment to run TensorFlow via Python. Next:

```
(tensorflow_p36) ubuntu@ip-172-16-0-163:  python
```

This will fire up Python. At this point, you will want to run the code provided. This code implements an RNN (a classic RNN that does not use LSTM) that tries to determine what file each line of text came from, by only looking at the sequence of characters. Look over the code before you run it. The supplied code will first load up data from the three files, and then it will run the backprop learning algorithm for 10,000 iterations, where each iteration processes a “mini-batch” that is a set of 100 randomly-selected lines from the three files. Learning (all 10,000 iterations) might take 10 minutes on the p2.xlarge machine.

Once you’ve run the learning algorithm (or after you’ve gotten sick of watching it for a few minutes), your task is to slightly modify the Python code I’ve provided so that at the end of learning, it says, “Average loss for the last 10 mini-batches is 0.9982, average correct labels is 56.5 out of 100”. Naturally, these values should not be hard-coded, they should be computed to represent the actual observed averages.

Once you have made this modification, run the code for the full 10,000 iterations. To get checked off, show the output for the last 20 iterations, plus the message that you added, to a TA/Instructor.