Journal Report 6

10/7/19-10/11/19
Odin Woitek
Computer Systems Research Lab
Period 1, White

Daily Log

Monday October 7

Started following a tutorial for making a Convolutional Neural Network. This involved training a network to be able to distinguish digits from the MNIST dataset. Everything worked pretty well except for adding the input layer, since the syntax from the tutorial was giving me an error because it yielded a negative dimension.

Tuesday October 8

I looked into the error I got yesterday. It turns out that Tensorflow requires you to specify that the data format is ”channels first” when the number of convolution filters being used is specified before the rows and columns of the convolution kernel. Once I had MNIST working, I started to work on creating the network I will use for my project, but immediately ran into trouble when dealing with loading data into the training and test sets. With MNIST, I could just load the data from keras.datasets.mnist, but I will need to load my own data myself. I began researching exactly how the training and testing sets are formatted and how they work, so I could plan out how I will load them.

Thursday October 10

I worked on getting the data loaded into training, testing, and validation sets. I started by going back to the code I wrote to generate the folder that contains all of my organized data and thought I might adjust that to create folders for training, test, and validation. But by doing that, I would have a ton of subfolders that are basically just smaller versions of the subfolders I have now, which seemed really messy, so I looked for another way to do it. I found that the scikit-learn library makes splitting data into training, test, and validation sets really easy, so I installed that and used it to split my data. I used glob to gather my data and organize it from my data folder, creating a numpy array for all inputs and a numpy array for all outputs. I used scikit-learn to split these arrays into a training set and a test set with a 5:1. I then used sci-kit-learn again to split the training set into the training set and the validation set with a 4:1 ratio. As a result the final ratio of train to test to validation was 4:1:1.

Timeline

Date	Goal	Met
September 23 - 27	Get organized data for top 20 writers	Yes, created data folder containing folders for each writer with the image patches we cropped from their writings.
September 30 - October 4	Set up Tensorflow for GPU for higher performance	No, Tensorflow GPU only works with NVIDIA graphics cards. I also tried optimizing for Intel chips but ran into problems there as well.
October 7 - 11	Load input and output data into training and testing sets	Yes, Used sci-kit learn to split data into training, testing, and validation sets.
October 14-18	Run and evaluate model	
October 21-25	Adjust model architecture to better fit out problem	

Reflection

This week was mostly just getting accustomed to the keras library and working to gain a better understanding of convolutional neural networks. I started out by following a simple tutorial that tackled the MNIST problem using a CNN, which started to give me an idea of what exactly I will need to do when I create the model for my handwriting identification problem. I did run into one syntax issue with the MNIST tutorial because they were using Theano as a backend and when creating the first convolutional layer of the network, specified the number of convolution filters being used before the rows and columns of the convolution kernel. In Tensorflow, you need to specify data_format = "channels first" if you specify the parameters in this order, which caused me some problems. Afterwards, I had to load the data into training, testing, and validation sets. Since the MNIST example just used the MNIST library in Keras, it didn't really teach me how to load the data, and I began to think of a way to reorganize my data into testing and training, but that would basically make a bunch of folders with the same subfolders with different parts of the data, since the data output was the name of the folder that the data was in. That seemed to complicated to deal with, so I looked for a different way to load the data and I found sci-kit learn, which made loading the data really easy, as shown here:

```
import numpy as np
import glob
from sklearn.model_selection import train_test_split

x = []
y = []
for direct in glob.iglob('data/*/'):
    author = direct.split('\\')[1]
    for filepath in glob.iglob(direct + '*.png'):
        x.append(filepath)
        y.append(author)
```

```
x = np.asarray(x)
y = np.asarray(y)

X_train, X_test, y_train, y_test = train_test_split(x, y,
                                        test_size=0.167, random_state=1)

X_train, X_val, y_train, y_val = train_test_split(X_train, y_train,
                                        test_size=0.2, random_state=1)
```