CMSC 25025 / STAT 37601

Machine Learning and Large Scale Data Analysis

Assignment 4 Due: Tuesday, May 14, 2018 at 2:00 pm.

Please hand in this Homework in 4 files:
1. A pdf of your jupyter notebook for problem 1.
2. The ipynb file for problem 1.
3. A pdf of your jupyter notebook for problem 2.
4. The ipynb file for problem 2.

We will post instructions on how to use RCC for tensorflow in an announcement. If you want to install it on your computer you should do it in a virtual environment here's how, it takes a few minutes.

```
conda create -n myenv tensorflow
conda install -n myenv ipython
conda install -n myenv jupyter
conda install -n myenv numpy
conda install -n myenv scipy
conda install -n myenv matplotlib
conda install -n myenv scikit-image
```

This will create a python 3 installation with tensor flow. Once the installation is complete

On your laptop:
Type: `source activate myenv`. Then at the prompt call the jupyter notebook.

1. *Convolutional networks for MNIST* (60 points)

   The code in `conv_net.ipynb` allows you to train a particular convolutional neural network (which we call the *original model*) on MNIST data. It also saves the model in your directory and has code to reload the model and continue training or to simply test the model on a new data set.

   (a) Compute the total number of parameters in the original model. And run this model. You shouldn't run more than 20 epochs. (On the RCC with 8 cores it takes about 90 seconds per epoch with all the training data.) You can do this with only 10000 training data to expedite the experiments. For each experiment plot the error rate on training and validation as a function of the epoch number.

   Show an image with the 32 5×5 filters that are estimated in the first layer of the model

   (b) Experiment with changing parameters of the network:

i. Keep the same number of layers and change layer parameters reducing number of parameters by half and doubling the number parameters. Try a few different options. Report the results.
ii. Design a deeper network with the same number of parameters as the original network. Report the results.
iii. Once you pick the best configuration try it on the full training set and report the result

(c) Handling variability. A transformed data set `/project/cmsc25025/mnist/MNIST_TRANSFOR` has been created by taking each digit, rotating it by a random angle between [-40,-20] or [20,40], applying a random shift of $+/-$ 3 pixels in each direction and applying a random scale between $[.9, 1.1]$.

Display a few of these examples alongside the original digits.

Using the original architecture to test on this data set. The classification rate drops dramatically.

Try to propose changes to the network architecture so that still training on the **original training** set you would perform better on the transformed test set. Perform some experiments using a transformed validation set and show the final results on the transformed test set.

2. *Convolutional networks for CIFAR10* (40 points)

(a) Read in the cifar data. The files are in `/project/cmsc25025/mnist/`. Display some of the images.

(b) Modify the code to apply the original network to the cifar data. Remember that the images now have 3 color channels. Again plot training and validation error against epoch number. Plot the first layer filters.

(c) Try to define a deeper network with the same number of parameters and see if you get an improvement.

(d) Variability. Use `skimage.color.rgb2hsv` to transform the rgb color map of the input images to and hsv - hue, saturation, value. You can use `hsv2rgb` to transform back. (To read more about different color coding methods see: `https://en.wikipedia.org/wiki/HSL_and_HSV`. Saturation is a value between 0 and 1 providing a sense of how ‘colorful’ the pixel is.) For each image in the test set multiply the saturation of all pixels values by a fixed value drawn randomly between .75 and 1.25. Then convert back to RGB. Show some of the resulting images. Run the model on the modified data and report the result.