ECE532 Final Project Proposal

Timothy Winfree

October 2020

1 The Dataset

For this project, I will be using the MNIST dataset located here: `http://yann.lecun.com/exdb/mnist/`. This dataset contains 70,000 20x20 binary images of handwritten digits 0-9. Each image is labeled with the integer corresponding to the handwritten digit. The classification task at hand is to create an algorithm that takes an image of a handwritten digit as an input, and outputs the corresponding integer as a label.

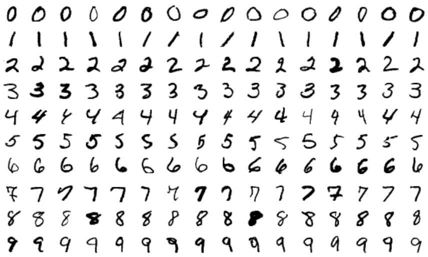

Figure 1: Example images of handwritten digits from the MNIST dataset

2 The Algorithms

2.1 Linear Regression: Least Squares, Regularization

Classification via linear regression requires user defined features. To create my features, I plan on designing a couple dozen or so structural correlation filters that I will convolve with each image. I will use the value and general position of the maximum of the filtered image as elements in a feature vector. For example, I can convolve each image with a 5x5 binary matrix with ones in the third row. If I have a relatively large max value located in the upper-middle part of the filtered image, I can guess I'm looking at a 7. With the features defined, the application of least squares and ridge regression is straight forward. I will play around with the ridge coefficient (lambda) during cross validation to determine which coefficient minimizes the expected error rate.

2.2 k-nearest neighbors

Using the same features discussed above, I will classify the images using the k-nearest neighbors algorithm. The k-nearest neighbors method classifies an image by examining a neighborhood around the image in feature space, and observing which of the k-nearest labels occurs most frequently. One can also add a weight function so that the labels of closer neighbors have more weight. During validation, I will play around with different weight functions and the value of k and compare error rates.

2.3 Convolutional Neural Networks

Convolution filters are a great method for extracting interesting features from images. Thus, the convolutional neural network lends itself nicely to solving this classification problem. I can use the same pattern matching filters I used for the previous algorithms to design convolutional layers, and use gradient descent to compute weights that minimize the cost function which I will have to define.

2.4 Validation

To perform cross validation of my classifiers, I will perform the following process a dozen or so times and average the error of each trial: Randomly choose 50,000 images from the training set to train the classifier, and use the remaining 10,000 images to measure the error rate. I can then train the classifiers using all 60,000 training images, and measure the number of miss-classifications on the 10,000 image test set and compare this error rate to the error rate predicted by cross validation.

3 Timeline

3.1 1st Update: Nov 17

By Nov 17th, I plan to complete feature extraction, implementation and analysis of the linear regression techniques, and implementation of the k-nearest neighbors algorithm.

3.2 2nd Update: Dec 1

By Dec 1st, I plan to have completed my analysis of the k-nearest neighbors algorithm and be close to halfway through my implementation of my 2-5 layer convolutional neural network.

3.3 Final Deadline: Dec 12

By the final deadline, I plan to complete the implementation of my CNN, and analyze the strengths and weaknesses of each of the three algorithms.

3.4 Link to Project

`https://github.com/twinfree/ECE532-Final-Project`