Paul Johnston

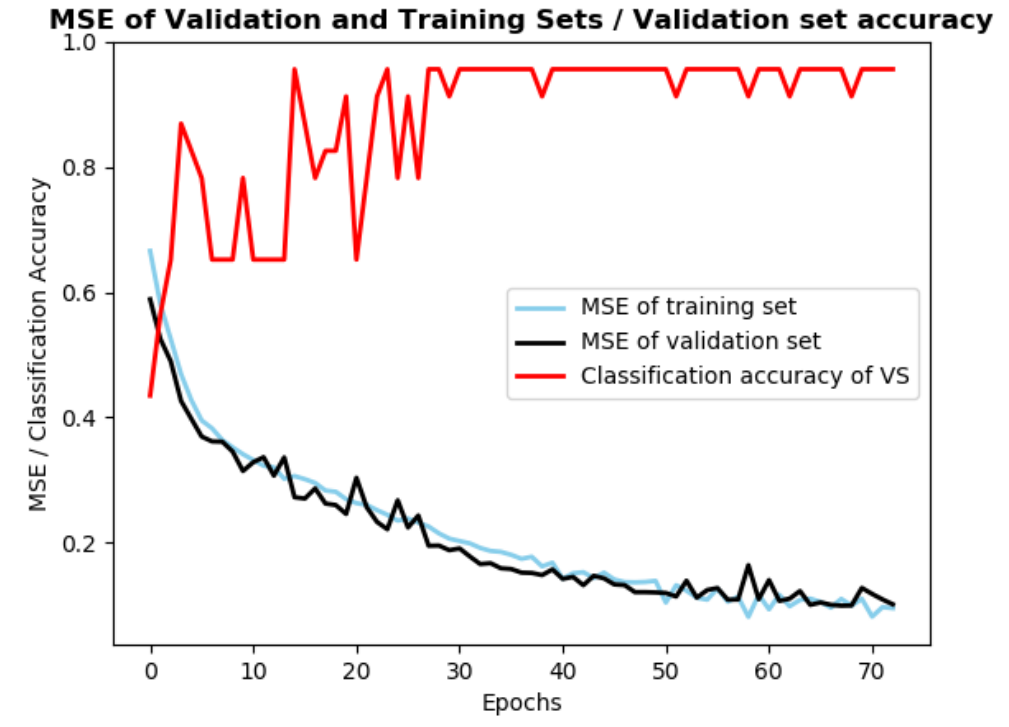
CS 478

MLP with Backpropagation

1. This MLP with Backpropagation (Backprop for short) has been designed to handle an arbitrary number of hidden layers with an arbitrary number of nodes for each layer. Initially, the weights between nodes are randomly set with random values between negative one and one, with a mean of 0. It uses stochastic weight updates meaning it changes the weights after each instance. It uses a non-linear activation function (in this case the sigmoid function) as well as the standard fprime(net) to determine error values for nodes. It separates out a validation set with which to test against at each epoch and determine if further epochs should be explored. It also uses an optional adaptive learning rate which is adding the change of a weight by the previous weights change multiplied by a "momentum" rate which can be set by the user. The stopping criterion is based on how many epochs the user is willing to explore with no decrease of the Mean Squared Error of the validation set.

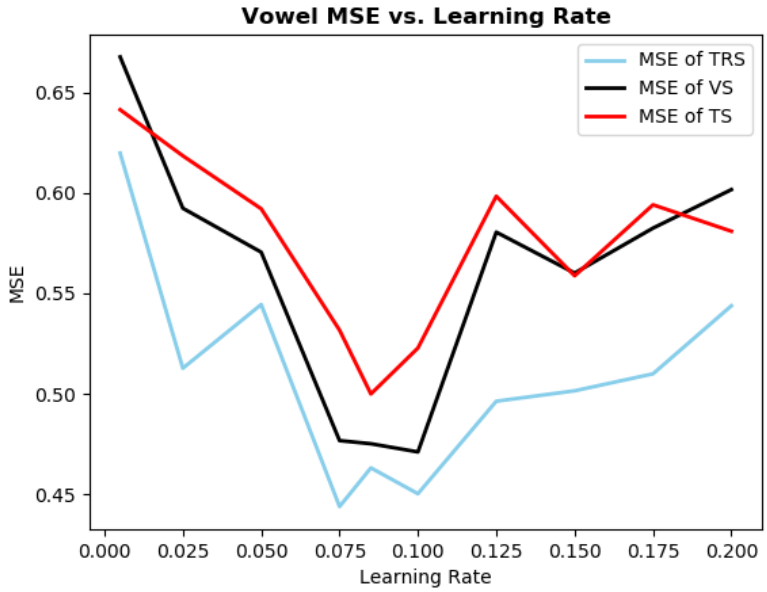
2. Iris Dataset

The graph below was produced using a learning rate of 0.1 and a momentum term of 0. There was one hidden layer with the number of nodes of the hidden layer being twice the number of nodes for the input layer. The data was split 75/25 into training/testing respectively. A validation set which was 20 % the size of the testing set was put aside for measuring accuracy at the end of each epoch. The final validation set accuracy was 96 percent, the final testing set accuracy was 95 percent. The MSE of the training set was computed for each epoch as well as the MSE for the validation set. Near the end of the epochs, the validation set accuracy was slightly declining while the training set accuracy was slightly increasing. This was expected as the network began to fit the training data more closely, while generalizing less, which is what the validation set measured. After 5 epochs of no improvement of the validation set MSE, the network stopped. Both the MSE's for the validation and training set are shown alongside the overall accuracy of the validation set as the epochs progressed. The scale of the y axis is consistent for both purposes.

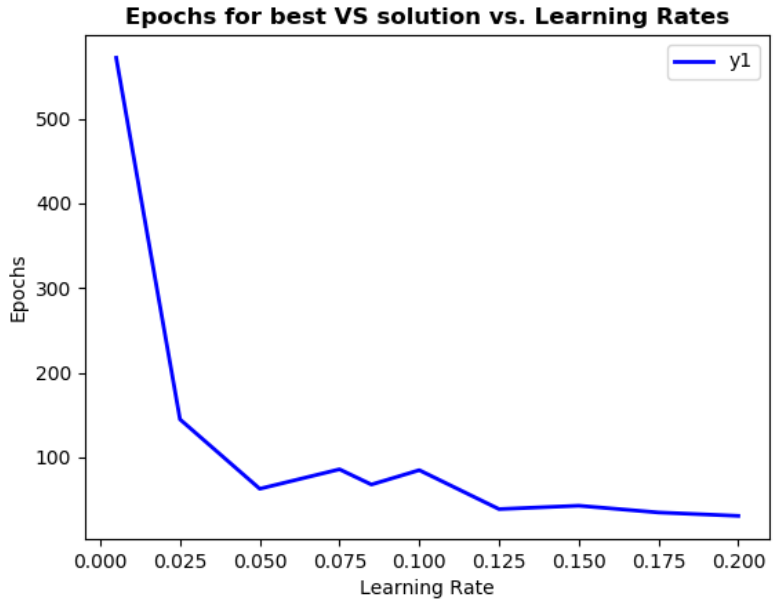


3. Vowel dataset

The baseline accuracy of this task would be about 9 percent with no machine learning involved, just random chance. The input features I chose to use were all but the first three: Train/test, speaker, sex. I didn't train/test as a useful feature because the manager would split up the data into training and testing portions regardless, the speaker name is not a general feature that will determine the vowel pronunciation, for example all Steve's don't have the same pronunciation. The sex feature I didn't see as useful because again, all males or all females don't have the same pronunciation among sexes. I kept the other ones because they seemed more a measurement of the actual pronunciation of the vowels. There was one hidden layer with twice as many nodes as the input layer. The data was randomly split 75/25 for training/testing and within that training set, a 80/20 split was made for training/validation. Several learning rates were applied which are shown in the graph below along with the corresponding MSE's for the training, validation, and testing sets. Each learning rate was tested once. Five epochs with no progress on the validation set MSE were allowed before stopping. The learning rate which resulted in both the highest testing accuracy and lowest validation set MSE was 0.085

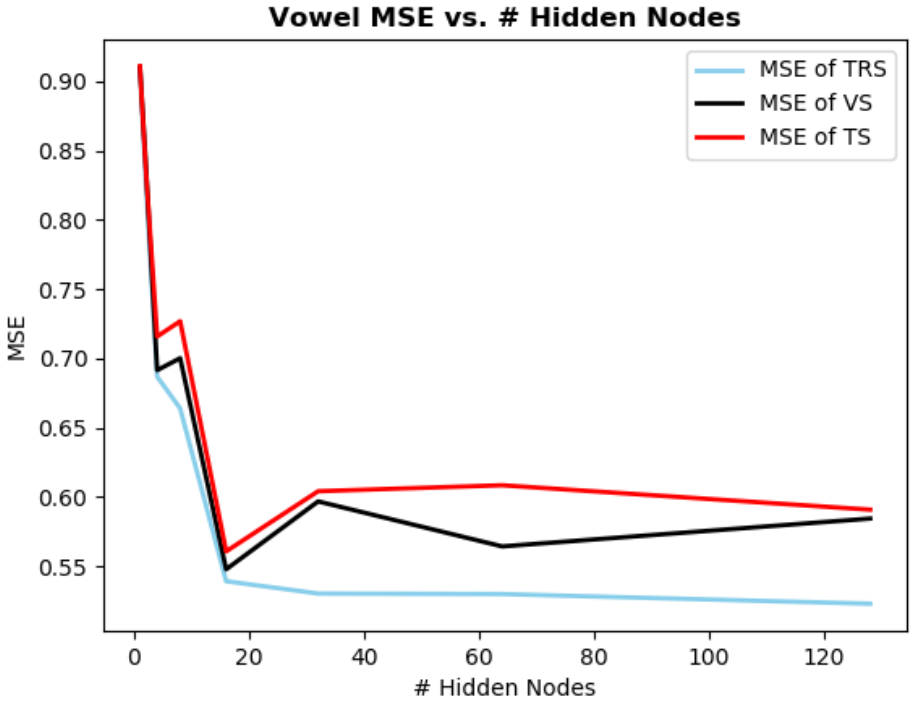


The graph below shows the number of epochs needed to reach the best validation set solution for each tested learning rate. The best learning rate took 68 epochs to finish.



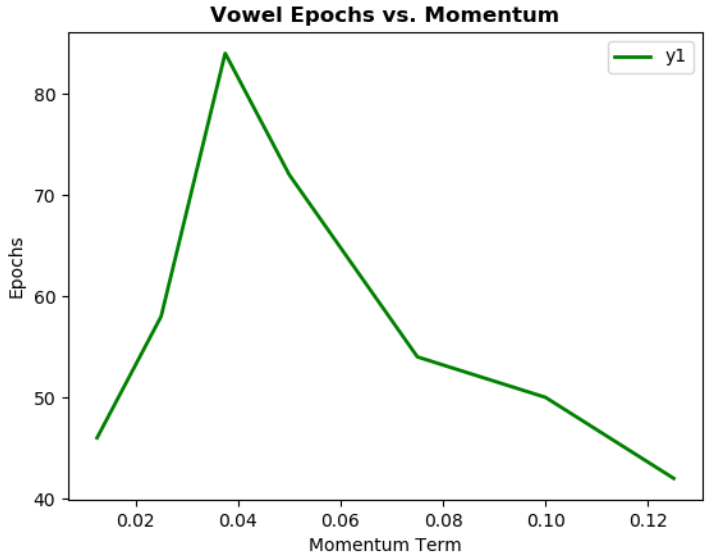
4. Hidden nodes

After using a learning rate of 0.085 from experiment number 3, the number of hidden nodes was experimented with. First, 1 hidden node was used, then 2, then double that, until no significant improvement was made. The best accuracy was attained by using 16 hidden nodes. The stopping criterion was checking the validation set MSE every epoch, and if no improvement was made for 5 epochs, it stopped. The accuracy of the test set was also greatest at 16 hidden nodes. This makes sense because the MSE of the sets also increased past this number of hidden nodes.



5. Momentum

With the learning rate set to 0.085 and number of hidden nodes set to 16, different momentum terms were tested and measured by how many epochs it took to find the best solution. At first, the number of epochs increased as the momentum term increased, but past 0.0375, the number of epochs began to decline. Also, the accuracy rate began to decline past this momentum term, so the number of epochs decreased, but accuracy did as well. This has led me to believe that although a higher momentum term allowed the model to learn faster, i.e. less epochs, it didn't learn as well. So the optimal momentum term for this model is 0.0375.



6. Experiment

For my own experiment, I tested whether multiple hidden layers improved accuracy. With the previously found optimal values for the learning rate of 0.085 and momentum term of 0.0375, I tested adding hidden layers of size 4 (including the bias node). After testing with 1, 2, 3, and 4 hidden layers, it was clear that the increase of layers past two was making the model less accurate. Also, with two hidden layers, the model trained for 107 epochs, the most out of any of the number of hidden layers tests. The accuracy at two layers was 33 percent, about half as accurate using just one layer with 16 nodes. To further my experiment and see if I could get even better accuracy using two layers, I made two layers with the hidden layer closest to the input layer being twice the number of input nodes and the hidden layer closest to the output nodes having half the number of nodes as the first hidden layer. The accuracy with these settings increased to 60 percent, about the same as it was with one layer. My conclusion is that until I have learned to utilize deep learning, adding multiple layers will not increase the overall accuracy dramatically.