3 Empirical Questions

The following questions should be completed after you work through the programming portion of this assignment.

For these questions, **use the small dataset**. Use the following values for the hyperparameters unless otherwise specified:

Parameter	**Value**
Number of Hidden Units	50
Weight Initialization	Random
Learning Rate	0.01

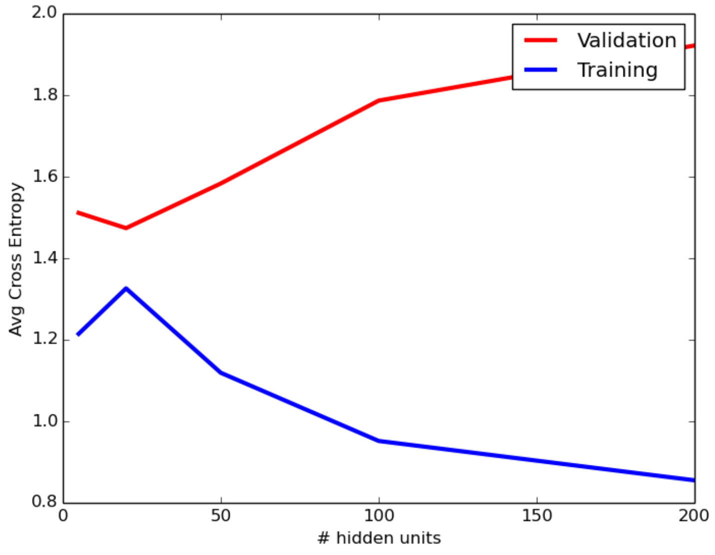
Please submit computer-generated plots for (a)i and (b)i. Note: we expect it to take about **5 minutes** to train each of these networks.

1. Hidden Units

   (a) (2 points) Train a single hidden layer neural network using the hyperparameters mentioned in the table above, except for the number of hidden units which should vary among 5, 20, 50, 100, and 200. Run the optimization for 100 epochs each time.

   Plot the average training cross-entropy (sum of the cross-entropy terms over the training dataset divided by the total number of training examples) on the y-axis vs number of hidden units on the x-axis. In the **same figure**, plot the average validation cross-entropy.

Avg. Train and Validation Cross-Entropy Loss



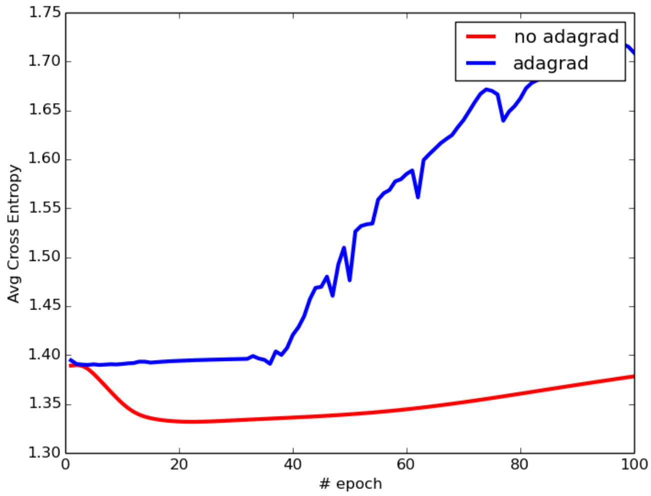
(b) (2 points) Examine and comment on the the plots of training and validation cross-entropy. What is the effect of changing the number of hidden units?

Answer

Training avg cross entropy decreases as hidden units increase, while validation avg cross entropy decreases then increase again. Additionally, training cross entropy is lower than validation due to better fitting of training data and due to overfitting in validation data. Generally as we increase hidden units, cross entropy gets lower for training, while validation cross entropy increases.

(c) (2 points) In the handout folder, we provide `val_loss_sgd_small.txt`, a text file with the validation cross-entropy loss values for SGD performed using 100 epochs, 50 hidden units, random init, and 0.01 learning rate. In the **same figure**, plot them against your validation results for SGD **with** Adagrad using the same set of parameters and the small dataset.

Avg. Validation Cross-Entropy Loss of SGD with and without AdaGrad



(d) (2 points) Examine and compare the two results. What do you observe?

Answer

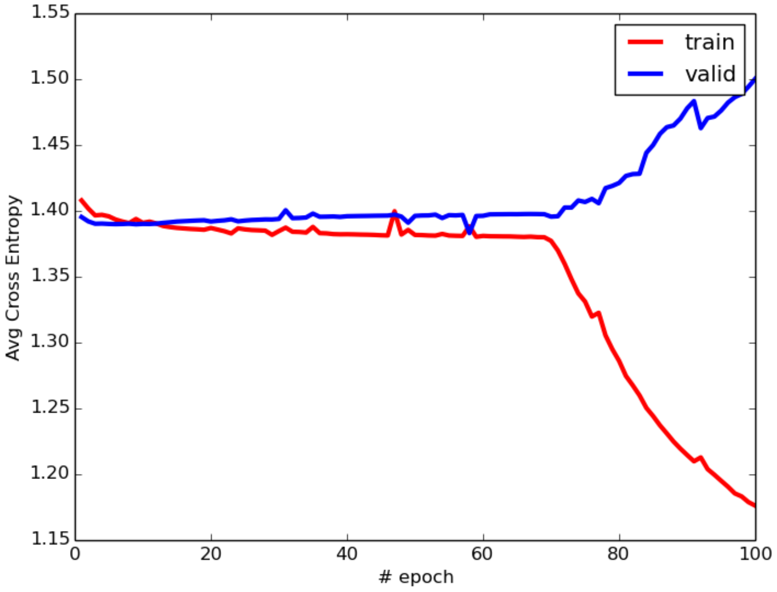
adagrad increases rapidly in avg cross entropy as epoch increases, while standard sgd without adagrad seems to decrease in the beginning # epochs but steadily increases at almost constant rate.

2. Learning Rate

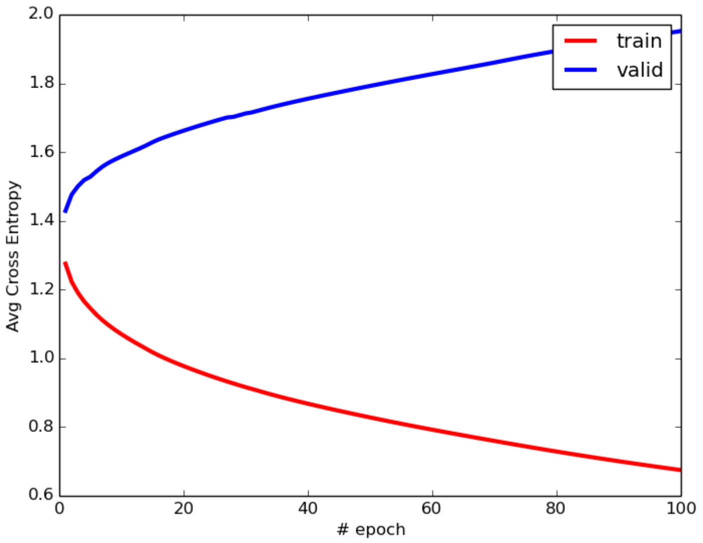
(a) (6 points) Train a single hidden layer neural network using the hyperparameters mentioned in the table above, except for the learning rate which should vary among 0.1, 0.01, and 0.001. Run the optimization for 100 epochs each time.

Plot the average training cross-entropy on the y-axis vs the number of epochs on the x-axis for the mentioned learning rates. In the **same figure**, plot the average validation cross-entropy loss. Make a separate figure for each learning rate.

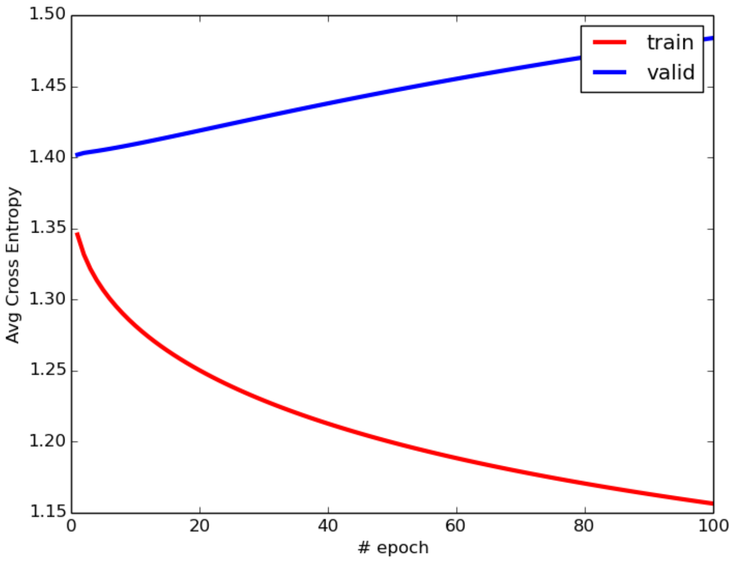
Plot LR 0.1



Plot LR 0.01



Plot LR 0.001



(b) (2 points) Examine and comment on the plots of training and validation cross-entropy. How does adjusting the learning rate affect the convergence of cross-entropy on the datasets?

Answer

validation loss goes away from zero in all cases of lr, increasing as the # of epochs increases. as learning rate decreases, the rate at which training cross entropy converges becomes quicker generally. however, lr=0.01 seems to decrease faster than lr=0.001, implying that 0.01 might be a better fit. Generally, decreasing learning rate seems to make the rate at which loss moves, whether closer or away from zero (goal), more exrreme.

3. Weight Initialization

(a) (2 points) For this exercise, you can work on any data set. Initialize $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ to zero and print them out after the first few updates. For example, you may use the following command to begin:

```
$ python neuralnet.py smallTrain.csv smallValidation.csv \
smallTrain_out.labels smallValidation_out.labels \
smallMetrics_out.txt 1 4 2 0.1
```

Compare the values across rows and columns in $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. Comment on what you observed. Do you think it is reasonable to use zero initialization? Why or why not?

Answer

Alpha values had same values in each row while beta values had same values across columns except for first column. No, it is not reasonable because same values of weights applies for multiple data values, meaning that it would underfit and cause the algorithm to not be as accurate and converge at slower rate.