Homework IV

STAT 6910/7810-002 - Spring semester 2019

Due: Friday, March 1, 2019 - 5:00 PM

Please put all relevant files & solutions into a single folder titled `<lastname and initials>_assignment4` and then zip that folder into a single zip file titled `<lastname and initials>_assignment4.zip`, e.g. for a student named Tom Marvolo Riddle, `riddletm_assignment4.zip`. Include a single PDF titled `<lastname and initals>_assignment4.pdf` and any Python scripts specified. Any requested plots should be sufficiently labeled for full points.

Unless otherwise stated, programming assignments should use built-in functions in Python, Tensorflow, and PyTorch. In general, you may use the `scipy` stack [1]; however, exercises are designed to emphasize the nuances of machine learning and deep learning algorithms - if a function exists that trivially solves an entire problem, please consult with the TA before using it.

Problem 1

1. It can be difficult at first to remember the respective roles of the ys and the as for cross-entropy. It's easy to get confused about whether the right form is $-[ylna + (1-y)ln(1-a)]$ or $-[alny + (1-a)ln(1-y)]$. What happens to the second of these expressions when y=0 or 1? Does this problem afflict the first expression? Why or why not? (Nielsen book, chapter 3)

2. Show that the cross-entropy is still minimized when $\sigma(z) = y$ for all training inputs (i.e. even when $y \in (0,1)$). When this is the case the cross-entropy has the value: $C = -\frac{1}{n}\sum_x [ylny + (1-y)ln(1-y)]$ (Nielsen book, chapter 3)

3. Given the network in Figure 1, calculate the derivatives of the cost with respect to the weights and the biases and the backpropagation error equations (i.e. δ^l for each layer l) for the first iteration using the cross-entropy cost function. Initial weights are colored in red, initial biases are colored in orange, the training inputs and desired outputs are in blue. This problem aims to optimize the weights and biases through backpropagation to make the network output the desired results. More specifically, given inputs 0.05 and 0.10, the neural network is supposed to output 0.01 and 0.99 after many iterations.

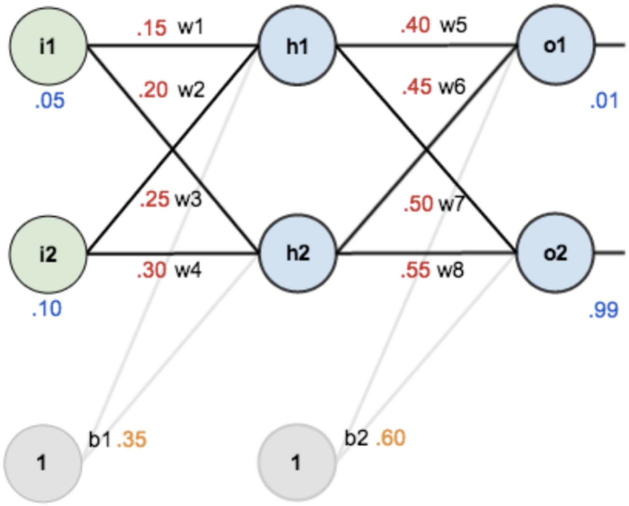

Figure 1: Simple neural network with initial weights and biases.

Problem 2

1. Backpropagation with a single modified neuron (Nielsen book, chapter 2)
   Suppose we modify a single neuron in a feedforward network so that the output from the neuron is given by $f(\sum_j w_j x_j + b)$, where f is some function other than the sigmoid. How should we modify the backpropagation algorithm in this case?

2. Backpropagation with softmax and the log-likelihood cost (Nielsen book, chapter 3)
   To apply the backpropagation algorithm for a network containing sigmoid layers to a network with a softmax layer, we need to figure out an expression for the error $\delta_j^L = \partial C / \partial z_j^L$ in the final layer. Show that a suitable expression is: $\delta_j^L = a_j^L - y_j$.

3. Where does the softmax name come from? (Nielsen book, chapter 3)

4. Show that with the log-likelihood cost and the softmax output layer,

$$\frac{\partial C}{\partial w_{jk}^L} = a_k^{(L-1)}(a_j^L - y_j).$$

Problem 3

1. Using either Tensorflow or PyTorch, design a single neural network for classifying the MNIST dataset. The neural network must have 2 hidden layers. In other words, your final neural network will have 4 layers total: the input layer, 2 hidden layers, and then the output layer. You will need to select the exact number of nodes in the hidden layers by tuning. However, you shouldn't choose less than 30

nodes or more than 100 nodes in each of the hidden layers. Use dropout and L2 regularization on the weights when training the network. Describe your entire design procedure. In particular, make sure to report the following:

(a) The dropout rate (i.e. the percentage of nodes dropped out each time), the activation functions in each layer, cost function, weight initialization strategy, and stopping criterion. These do not need to be tuned but you should provide some justification for each choice.

(b) Learning rate, regularization parameter, mini-batch size, and the number of nodes in each of the hidden layers. Each of these should be tuned in some fashion using a validation data set. Describe your tuning procedure for these parameters.

(c) The final test error. To get full credit for this problem, you will need to obtain a test accuracy greater than 98% as this was the accuracy obtained using a single hidden layer with regularization. Partial credit will be awarded for designs that do not perform as well.

(d) Include all of your code.

2. Starting with the network you trained in the previous problem, replace L2 regularization with L1 regularization and tune the regularization parameter as well as the learning rate. Use two initialization strategies: 1) initialize with the weights obtained using L2 regularization and 2) initialize randomly. Which initialization strategy worked the best? Based on your results, which regularization worked best on this data?

   Resources: There are multiple online resources for getting started with either Tensorflow or PyTorch. In addition, the TA has posted some basic PyTorch code in the file digit_classifier.py in the Python Bootcamp folder on Canvas. This may be helpful.

   Remarks: This problem will account for somewhere between 40-50% of the grade of this assignment. Accordingly, it may take you some time. I suggest you start early.

   For grading, we will be considering: 1) the final performance of your classifier, 2) whether your network fits the design criteria and whither you reported all of the requested information, and 3) whether your design choices make sense based on what we've talked about in class. For example, we have talked about multiple cost functions for classification in class. Not all cost functions will work as well for this problem. If you use a cost function that is not as well-recommended for this problem, you will receive a lower grade and have a harder time training.

   You may want to start by training a neural network with a single hidden layer. In class, we have even come up with good parameters for this case. You can use this as a starting point.

Problem 4

Did you fill out the midterm evaluation? A link will be posted in the assignment. A yes answer gets you about 2 points. No will give you zero. Lying may result in -2 pts for everyone.

References

[1] “The scipy stack specification.” [Online]. Available: https://www.scipy.org/stackspec.html