HW9: Feedforward NN
Due: 11pm on March 9, 2021

The assignment has two parts:

- Q1-Q2: Please read chapter 1 of NN book at `http://neuralnetworksanddeeplearning.com/chap1.html` and answer the questions. You can also check the hw9 code to get a more concrete idea of how feedforward NN works.
- Q3-Q5: Check an implementation of feedforward NN, and test its performance with respect to different hyperparameters and activation functions. Add a few lines of code to implement tanh as an activation function.

The hw9 code is under dropbox/20-21/572/hw9/code/:

- hw9.sh: the shell script that calls hw9_script.py.
  - To run hw9.sh, the command line is "./hw9.sh config_file output_file"
  - config_file specifies the hyperparameters of the NN (e.g., **config1.yml**)
  - output_file will be a file consisting of "# correct / # instances" per epoch. For the test accuracy in Table 1 and 2, just run hw9.sh with a config_file and report the final line as a percentage.
  - The script will print runtime to stdout.
- hw9_script.py: the python script that reads in the training data and test set, and call network.py to do training and testing.
- network.py: the code that defines the architecture of the NN, including the activation function and its derivative.

Q1 (15 points): Suppose a feedforward NN (sometimes called multilayer perceptron or MLP) has m layers: the input layer is the 1st layer, the output layer is the last layer, and there are $m-2$ hidden layers in between. The number of neurons in the k^{th} layer is n_k. Each neuron in one layer is connected to every neuron in the next layer and there are no other connections. Let's ignore the bias for Q1. That is, the model only has weight w, and there is no bias b.

(a) 5 pts: How many model parameters (i.e., weights) are there in this network?

(b) 10 pts: Let x be a column vector[1] that denotes the values of the input layer. Let M_k denote the weight matrix between layer k and $k+1$; that is, the cell $a_{i,j}$ in M_k stores the weight on the arc from the j^{th} neuron in layer k to the i^{th} neuron in layer $k+1$. Let's assume all the layers use

[1] A row vector is a $1 \times n$ matrix (e.g., $[a_1, a_2, ..., a_n]$); a column vector is a $n \times 1$ matrix. If you transpose a row vector, you get a column vector. For more info, see `https://en.wikipedia.org/wiki/Row_and_column_vectors`

the same activation function g, which is a function from R to R (R is the set of real numbers). If x is a column vector, $g(x)$ will apply the function g to every element in x and return a column vector as the output.

- Given the input x (x is a column vector), what is the formula for calculating the output of the first hidden layer?

- Given the input x, what is the formula for calculating the output of the output layer?

Hint: In class, we show the formula for calculating the z and y value for a neuron, where $z = b + \sum_j w_j x_j$ and $y = g(z)$. In Q1, let's assume b is zero.

Now let's look at layer i: the outputs of the neurons in that layer form a column vector of the size $n_i \times 1$. Let's call this vector y_i. The question is how one can calculate the output vector, y_{i+1}, at layer $i+1$ given the weight matrix M_k and y_k. Once you figure out how to do that, you can apply the same formula layer by layer, in order to get the output of the output layer given the input x.

Q2 (30 points): Suppose that you are training a neural network to do text classification, with $n > 2$ classes.

(a) 5 pts: What loss function can you use for the output layer? Can the loss function be the error rate on the training data? Why or why not? Here, let's assume that you need to use backpropagation for training.

(b) 15 pts: What are the main idea and benefit of stochastic gradient descent?

What is a training epoch?

Let T be the size of the training data (i.e., the number of training instances), m be the size of mini-batch, and your training process contains E training epoches. How many times is each weight in the NN updated?

(c) 10 pts: How can one choose the learning rate? What's the risk if the rate is too big? What's the risk if the rate is too small?

Q3 (15 points): Run hw9.sh with different config file settings (e.g., **config1.yml** and **config2.yml** are the config files for the first two experiments in Table 1). The **activation** value in the config file should be set to 0 (for sigmoid function). For the learning rate, keep it as 0.5. Fill out Table 1.

Table 1: Classification accuracy with **sigmoid** activation function

Expt id	# of hidden layer	# of neurons in hidden layers	# of epoches	mini-batch size	test accuracy	CPU time (in minutes)
1	1	30	30	10		
2	1	30	30	50		
3	1	30	100	10		
4	1	60	30	10		
5	2	30, 30	30	10		
6	2	40, 20	30	10		
7	3	20, 20, 20	30	10		

Q4 (20 points): Modify one or more python files in the hw9/code/ directory so that the new code will use tanh when **activation** value in the config file is set to 1. For the learning rate, keep it as 0.5.

- Fill out Table 2, which is the same as Table 1, except that it uses **tanh** as the activation function.
- In the readme.[txt | pdf], explain which functions (or which lines) in which file(s) you have changed.
- Submit the modified python code. Please keep the file names unchanged.

Table 2: Classification accuracy with **tanh** activation function

Expt id	# of hidden layer	# of neurons in hidden layers	# of epoches	mini-batch size	test accuracy	CPU time (in minutes)
1	1	30	30	10		
2	1	30	30	50		
3	1	30	100	10		
4	1	60	30	10		
5	2	30, 30	30	10		
6	2	40, 20	30	10		
7	3	20, 20, 20	30	10		

Q5 (20 points): Answer the following questions:

(a) 10 pts: For any experiment in Q3 or Q4, if you run it multiple times, you are likely to get different results. Why is that?

- specify the line numbers in the python code that causes this non-deterministic behavior of the system.

- Is this kind of behavior desireable? Why or why not?

(b) 10 pts: What can you conclude from Tables 1 and 2, and from Q5(a)?

Submission: Submit the following to Canvas:

- Your note file *readme.(txt | pdf)*, which includes your answers to Q1, Q2, Q4, Q5, and Tables 1-2, and any notes that you want the TA to read.
- hw.tar.gz that includes two python files specified in dropbox/20-21/572/hw9/submit-file-list.
- Make sure that you run **check_hw9.sh** before submitting your hw.tar.gz.