Name:Serena Yang
CSC 578
HW5

1. **Experiement results**

   The first eight experiments only changed with different act_hidden, act_output, and cost functions; I got the same results compared to the sample results. However, for the 9th experiment and 10th experiment, which are with the same act_hidden(Sigmoid), act_output(Sigmoid), and cost function(QuadraticCost) but different regularization(L1/L2), I got different results from the sample results. For the 9th experiment, even though the results are totally different, the results for the last few test epoch are really close to the sample result. For the 10th experiment, even though the first test epoch results are different from the sample result, the last epoch results' are the same. In my result, if we look closely, the output became stable a few epochs earlier than the sample results, around two epochs. Since the change only starts from the 9th and 10th experiment, we only made changes for the L1/L2 regularization in update_mini_batch() and total_cost() functions. I assume it might be because of learning rate selecting, lambda, mini_batch, and regularization. For the 11th experiment, the last epoch results are close; for the 12th experiment, the results are the same starting from the second epoch. The final results are pretty close with a particular dropout parameter, as shown for the 11th experiment result.

   9th experiment:

   Sample result

```
Test correct:
[18, 20, 38, 38, 38, 38, 38, 38, 38, 38, 39, 39, 42, 44, 47]

Training correct:
[30, 31, 62, 62, 62, 62, 62, 62, 62, 62, 62, 63, 66, 71, 74]
```

   My result

```
[38, 38, 38, 38, 38, 38, 38, 38, 38, 38, 41, 42, 44, 45],
[62, 62, 62, 62, 62, 62, 62, 62, 62, 62, 62, 63, 64, 66, 72])
```

   10th experiment:

   Sample result

```
[35, 20, 20, 20, 20, 20, 17, 17, 17, 17, 17, 17, 17, 17, 17]
[65, 30, 30, 30, 30, 30, 33, 33, 33, 33, 33, 33, 33, 33, 33]
```

   My result

```
[17, 20, 20, 0, 20, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17],
[33, 30, 30, 0, 30, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33])
```

   11th experiment:

   Sample result

```
[17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 34]
[33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 65]
```

   My result

```
[6, 17, 17, 17, 18, 41, 40, 40, 41, 41, 45, 48, 52, 35, 35],
[11, 33, 33, 33, 35, 66, 66, 65, 65, 67, 70, 75, 87, 65, 65])
```

12th experiment:

Sample result

```
[17, 17, 17, 17, 17, 18, 17, 17, 17, 17, 17, 35, 35, 17, 17]

[33, 33, 33, 33, 33, 32, 33, 33, 33, 33, 33, 65, 65, 33, 33]
```

My result

```
[20, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17],
[30, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33])
```

2. Implementation

I completed modification for the derivative() to return the first derivative of the CrossEntropy function in CrossEntropy class and wrote the LogLikelihood function to return the cost associated with an output "a" and desired output "y" and return the first derivative of the Loglikelihood function to get the cost function for question A1.
Also completed modification for the softmax to get the softmax of vector z and the derivative of the softmax function, and wrote Tanh and ReLU activation function with same two functions, fn(z) and derivative(cls,z), for nodes on all hidden layers for question A2. I also completed adding the activation function for the nodes in the output layer, changing the cost function to QuadraticCost and print a warning if the cost function is Tanh, in set_parameters() for the Network class for question A3.
Moreover, I completed regularization with 'L2' and 'L1' parameter options in update_mini_batch() and total_cost() for question A4 by adding two different calculations for weights and biases.
Lastly, for questions A5 and B, I completed adding the dropout function, which is the parameter that specifies the percentage of dropout for nodes in hidden layers, which can help the model prevent from the overfitting problem. Also, the value is between 0 to 1; for the 11th and 12th experiments, we used 10% and 50% as the percentage of dropouts. I wrote the codes to assume all the hidden layers used the same dropout percentage and the dropout nodes selection for each hidden layer by randomly setting a fraction rate of units to 0 every update. Also, in SGD() function, I added a dropout mask and applied the mask in hidden layers during the forward propagation phase. Furthermore, in the backprop() function, I applied the dropout into the hidden layer during the backward propagation phase.

3. Reflections

In this assignment, for the question from A1 – A4, the difficulty level was medium after applying the formulas from the NNDL book into python code. But for question A5, the difficulty is hard for me. I need to review the material from week 4 and week 5, and get a clear vision of the dropout for the entire mechanism for the model process. The challenge I am facing now is limited time for this course. Since this is my last semester, working on this course and time-series class while preparing for the interviews, the time is not enough for this challenging course. Even though I spent one more day finishing this homework, after I finished this assignment, I feel I got clearer on backpropagation Hyperparameters and neural networks.