Mikhail Skobov

Assignment 3

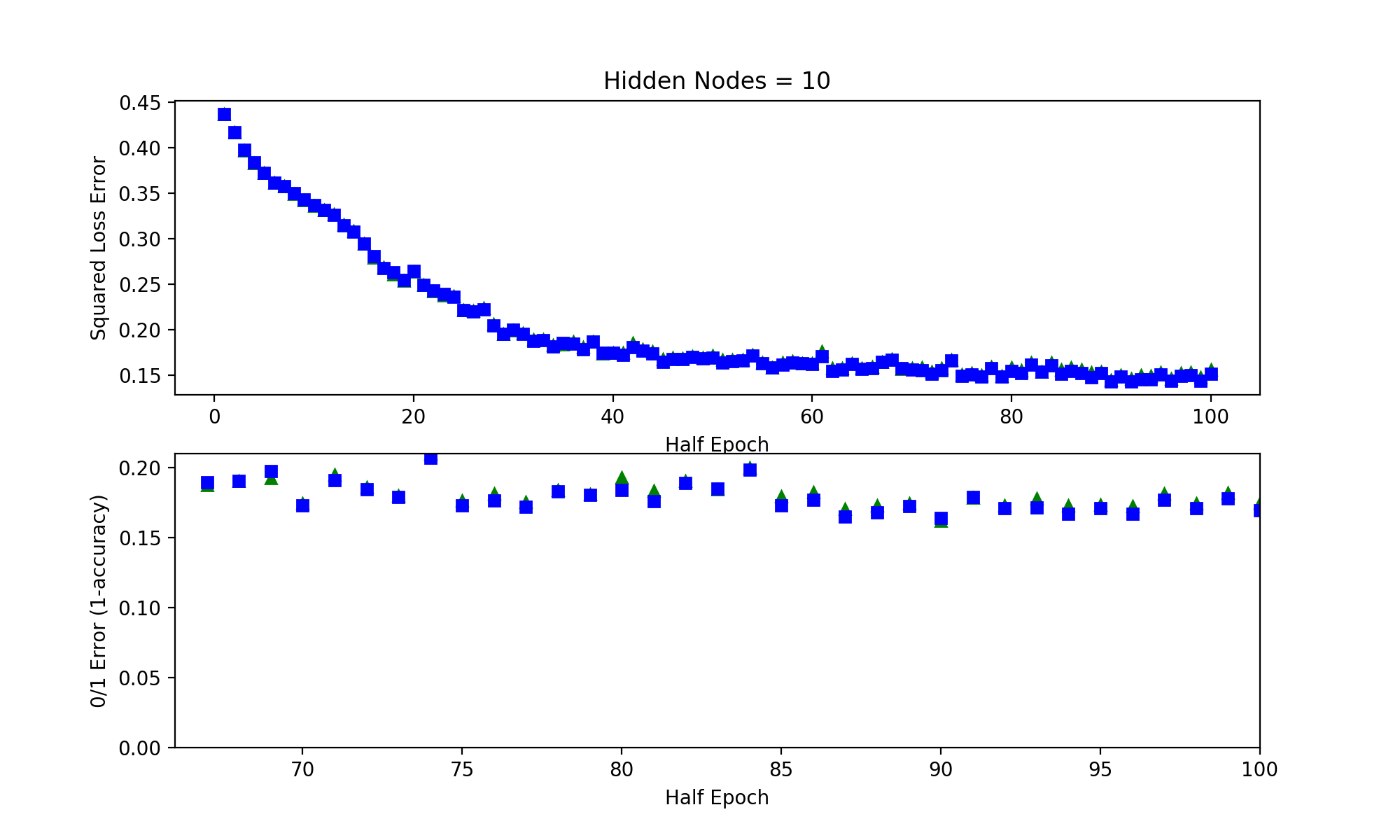
CSEP 546

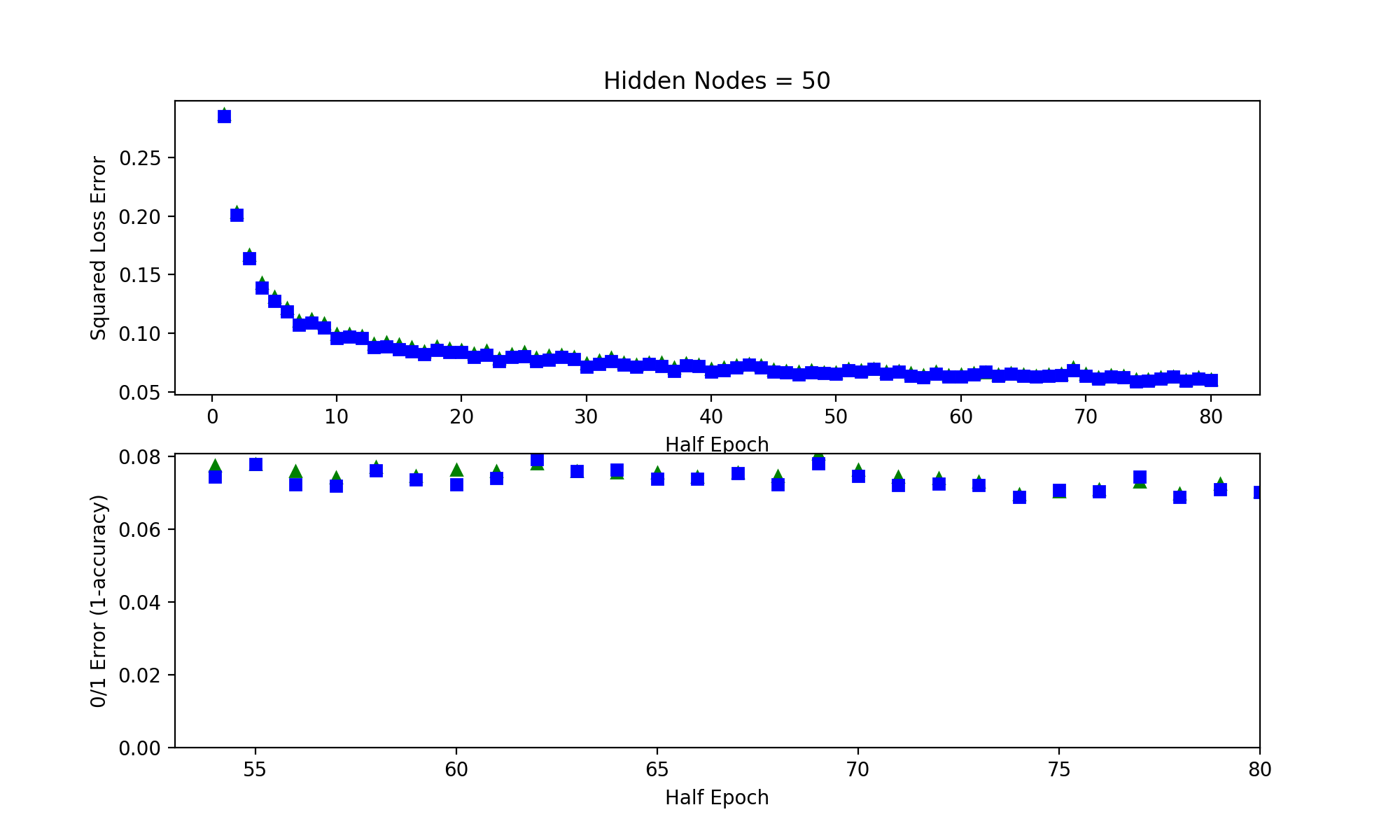
1.1 Write a paragraph describing your design choices. In particular, specify all your parameter choices: your learning rate (or learning rate scheme), mini-batch size, initialization scheme.

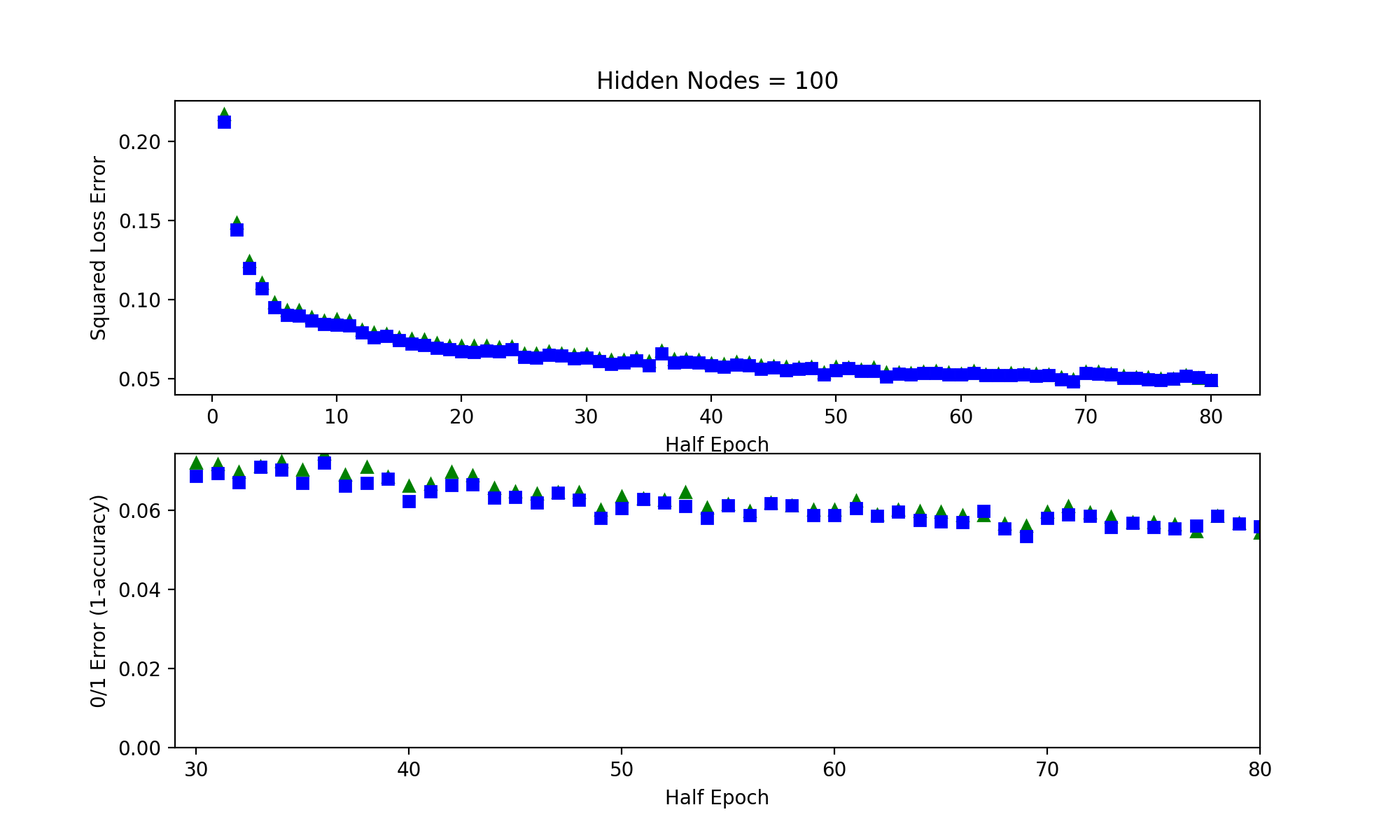
The parameters were effectively chosen by trial and error. The learning rates are set to their values because of convergence performance, and having a reasonable accuracy towards the end. Learning rate decay was not implemented, weight decay was, but it was not used. Mini batch sizes were either 50 or 100 for everything except ReLU. Higher values were chosen to speed up calculation. Initialization scheme was set to a normalized random distribution between positive and negative 2/sqrt(n) for sigmoids and sqrt(2/n) for ReLU. The table below shows all of the configurations.

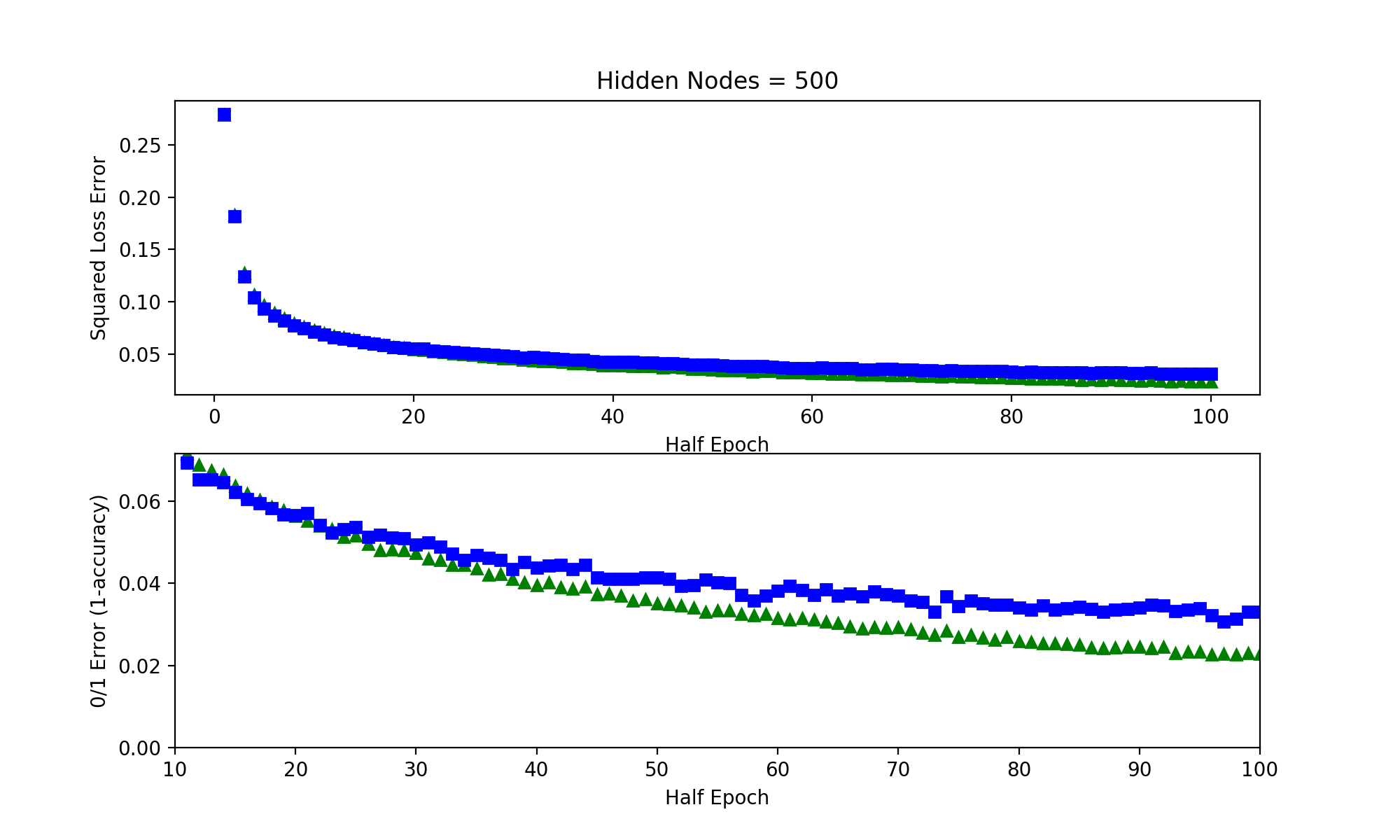
|  |  |  |  |
| --- | --- | --- | --- |
| Nh | Batch Size | Learning Rate | Momentum |
| 10 | 50 | .05 | .7 |
| 50 | 50 | .06 | .7 |
| 100 | 50 | .1 | .9 |
| 500 | 100 | .1 | .9 |
| 1000 | 100 | .08 | .9 |
| 500 (ReLU) | 1 | .0001 | 0 |

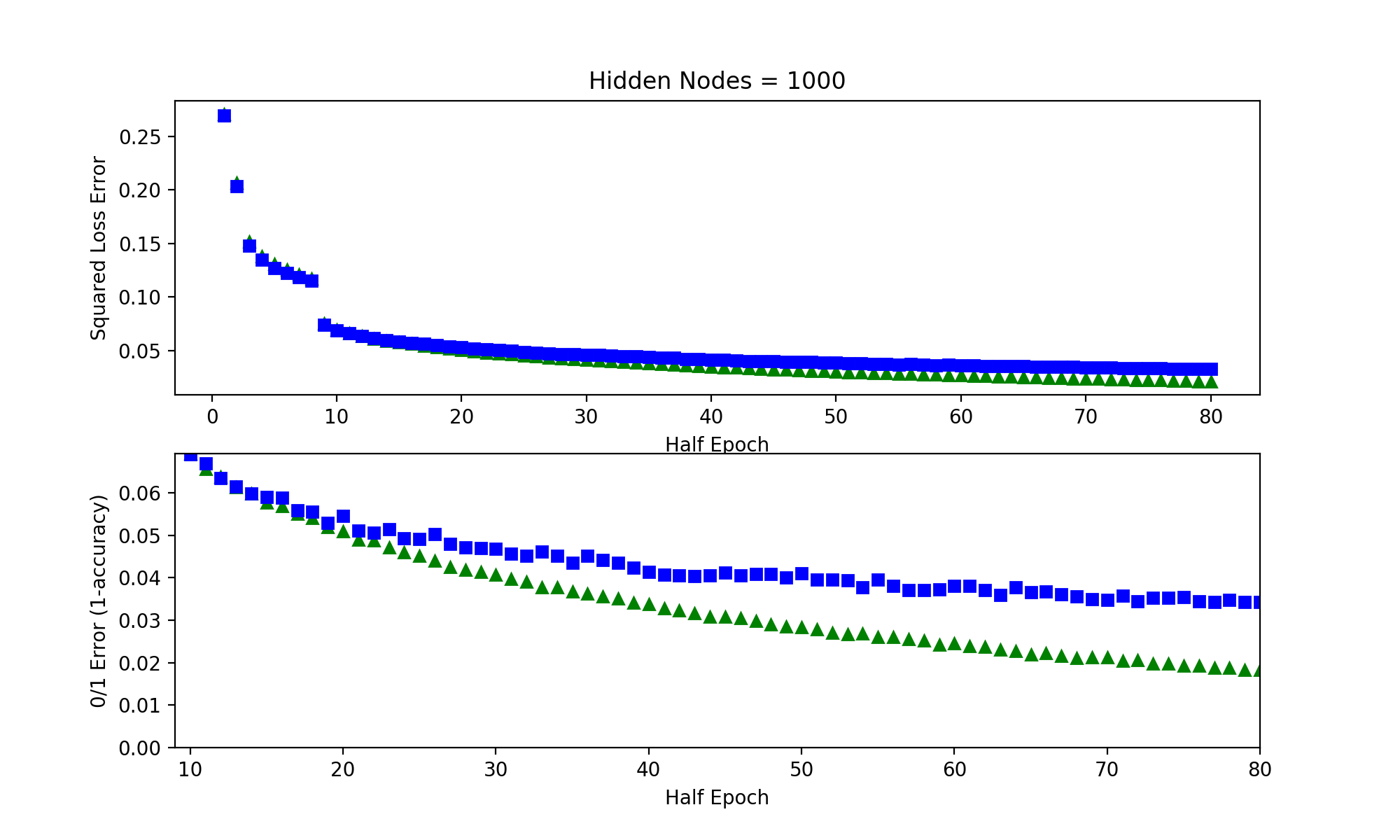
1.2,1.3 Please note that the blue squares represent test data and the green triangles represent the training data on all graphs.

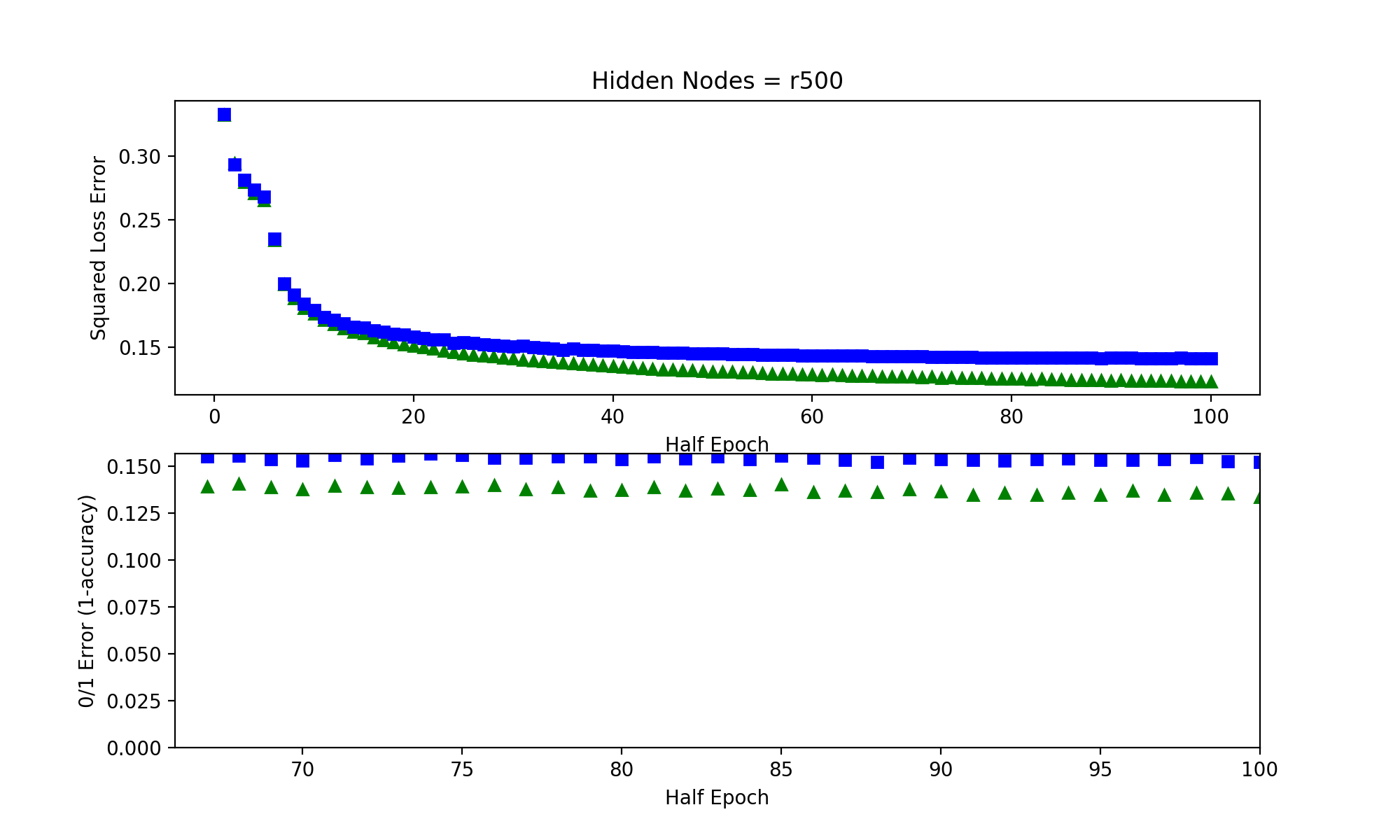










The graph below is for ReLU

1.4 What is your final squared loss and 0/1 loss for both the training and test sets for each network?

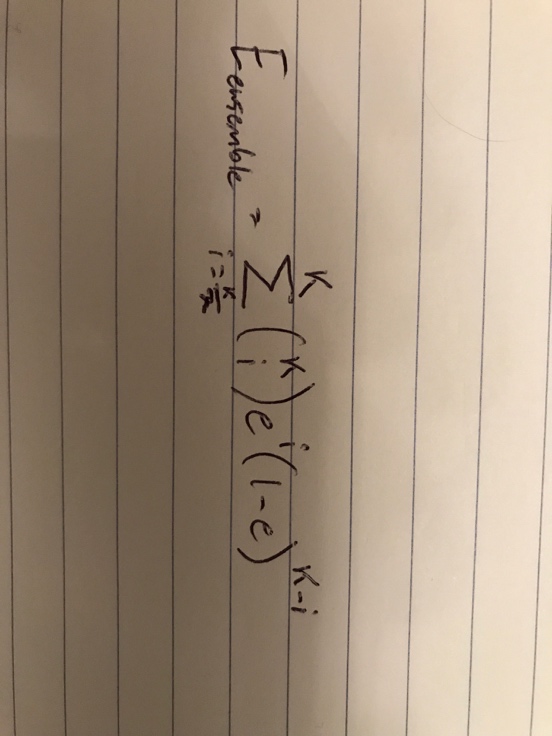
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Hidden Nodes | Training Sq Loss | Test Sq Loss | Training 0/1 | Test 0/1 |
| 10 | .156 | .155 | .175 | .169 |
| 50 | .061 | .06 | .0704 | .0702 |
| 100 | .0489 | .049 | .0543 | .0559 |
| 500 | .024 | .0315 | .0228 | .0331 |
| 1000 | .0213 | .0331 | .0183 | .0343 |
| 500 ReLU | .123 | .141 | .133 | .152 |

1.5 How does using ReLUs compare to using the sigmoid function? Why?

In the graph above, it shows that ReLU is performing a fair amount worse than the sigmoid. After analyzing the data (not shown) it appeared to have over-generalized, and set the weights low/off for the output of a value (it basically never guessed 7). Below is a description of other ReLU pros and cons:

ReLUs can have a tendency to overfit data. Since they make some nodes/weights go to 0, it essentially turns off a portion of the net, thus not allowing partial/small values to flow through. This creates sparsity, which can be good for a network (as opposed to a dense one). They can also have the opposite problem of “blowing up” where some weights become very large, and can over-exaggerate some results based off of certain weight paths. On the computational side of things, ReLU can be faster than evaluating the sigmoid function. One benefit of ReLUs is that they reduce and/or eliminate the vanishing gradient problem, which prevents a network from learning past a certain point, due to gradients not being large enough to significantly affect weights.

2.1



|  |  |  |  |
| --- | --- | --- | --- |
| e\K | K = 5, I = 3 | K = 10, I = 6 | K = 20, I = 11 |
| e = .1 | 0.00855 | 0.0001469 | 7.0886e-07 |
| e = .2 | 0.0576 | 0.0063693 | 0.0005634 |
| e = .4 | 0.3072 | 0.16613 | 0.12752 |

Note that “i” is the number of errors allowed. Assuming we want >50% agreement to accept hypothesis output.

2.2 Yes, it is possible. Shown via example: take K = 3, e >= .4.

|  |  |  |  |
| --- | --- | --- | --- |
| Hypothesis 1 | Hypothesis 2 | Hypothesis 3 | Result |
| Error | Error | Correct | Error |
| Error | Correct | Error | Error |
| Correct | Error | Error | Error |
| Correct | Correct | Correct | Correct |
| Correct | Correct | Correct | Correct |

Out of 5 examples, each hypothesis was wrong 2 times (2/5 = .4 = e). The overall error rate, however, is 3/5 = .6. Since they are no longer independent, we can say that any 2 hypotheses having an error together is now more likely, since they are correlated, which implies more results that are similar.