CS 838 Lab 2 Report

Chen Wang

February 18, 2017

1 Introduction

I implemented flexible hidden layers (but I just used single layer to do experiment for this assignment), both RELU and Sigmoid activation function, momentum and weight decay techniques, etc. The highest accuracy that my program can reach is around 62%.

2 Experiment

2.1 Different Number of Hidden Units

I did my first set of experiments on different number of hidden units. I used 10, 100 and 1000 hidden units. Other settings are the same. (Sigmoid activation function, learning rate = 0.01, momentum = 0.9, no weight decay).

The results are shown below, in Figure 1, 2 and 3.

From the result, we can see that with 100 hidden units, we get the best accuracy on testing set. 10 hidden units is also not a bad choice, but we can see that it doesn't has a best "early stop point". However, for 100 hidden units, we can see that the "early stopping point" is pretty much the best one we can choose. For 1000 hidden units, we can see that there is a heavy problem of over-fitting.

2.2 Sigmoid vs RELU

For this set of experiment, we have the exactly same settings with the last group, except for the activation function for this group is RELU. The results are shown in Figure 4, 5 and 6.

We can see that sigmoid has a better performance for all three situations.

2.3 Different learning rate

I used learning rate 0.1 and 0.25 comparing with our previous 0.01, with other settings remain the same. (sigmoid activation function). The results are shown in Figure 7 and 8.

It converges fastest when the learning rate is set to 0.1. When learning rate becomes 0.25, it takes a longer time to come to a stable state of tuning set accuracy. It is pretty slow for the network to converge when learning rate is 0.01.

2.4 Only with Weight Decay

This time, I set momentum to 0 and test weight decay for 0.01, 0.05 and 0.1. The results are shown below. The results are shown in Figure 9 and 10 and 11.

The results are really bad. It only outputs the label "_" for the prediction.

2.5 With both momentum and weight decay

I set momentum to 0.9 and weight decay to 0.05 this time. The result is Figure 12.

The result is really bad.

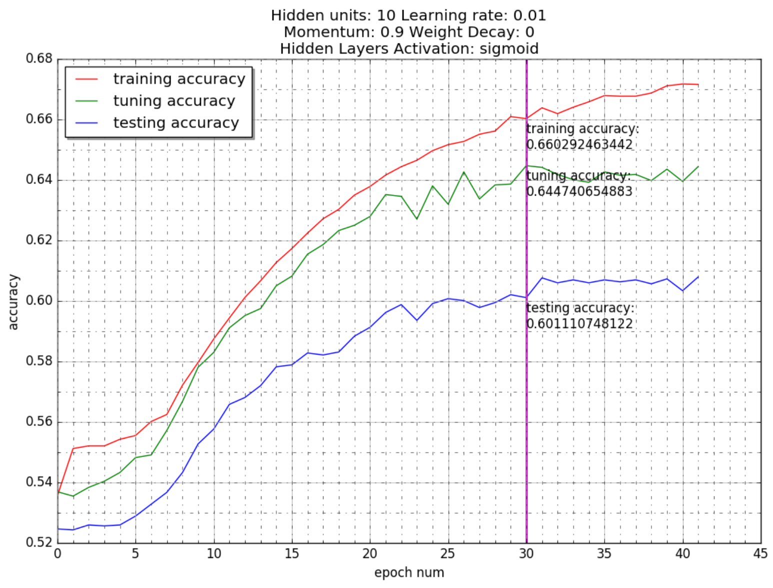

Figure 1: Hidden Units 10

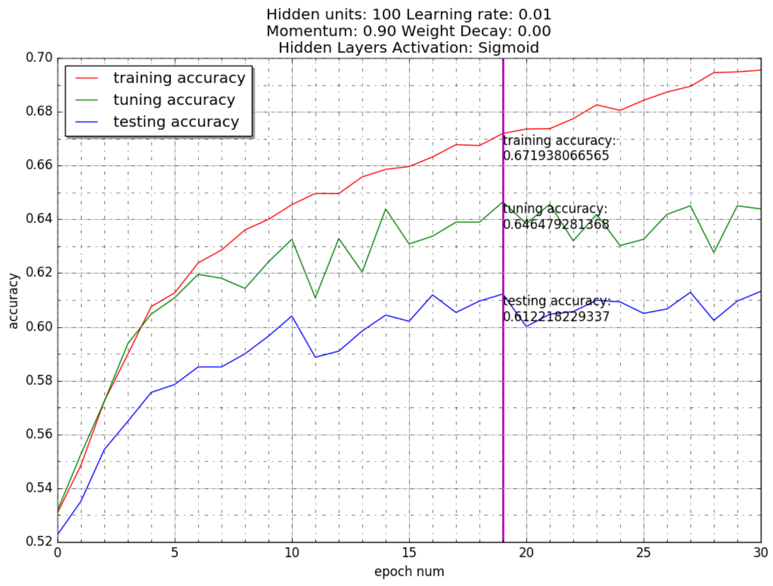

Figure 2: Hidden Units 100

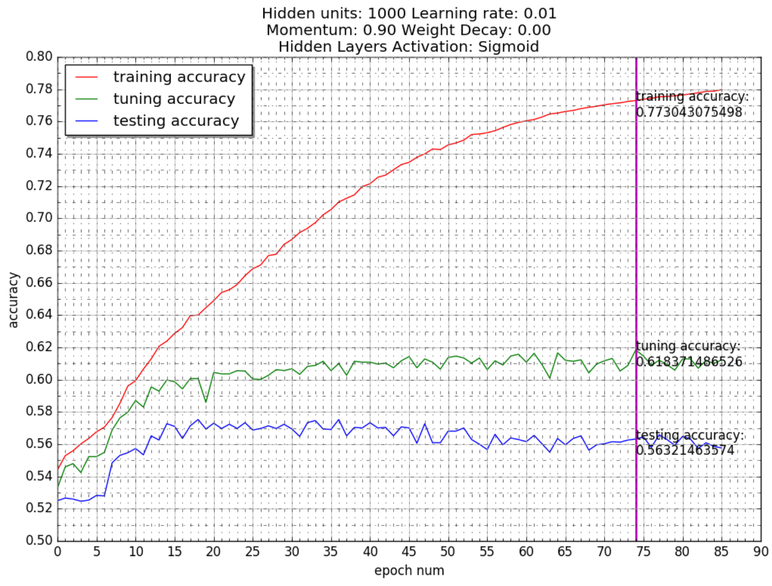

Figure 3: Hidden Units 1000

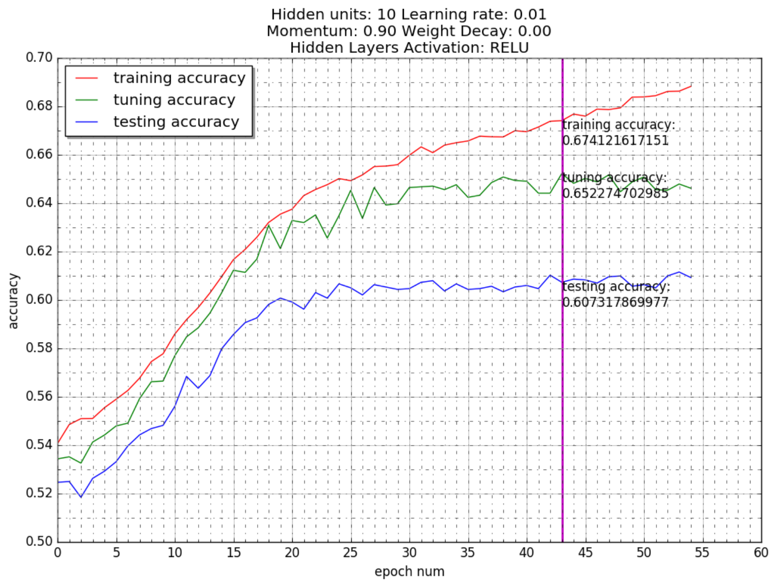

Figure 4: Hidden Units 10 RELU

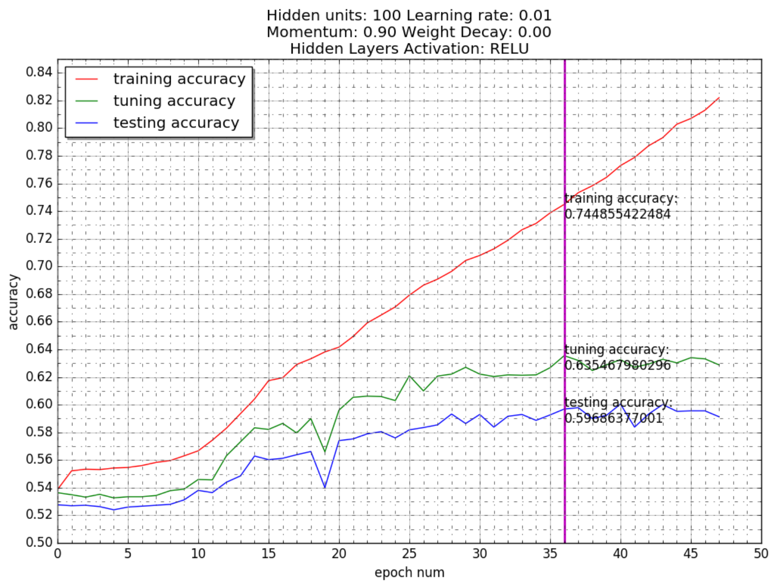

Figure 5: Hidden Units 100 RELU

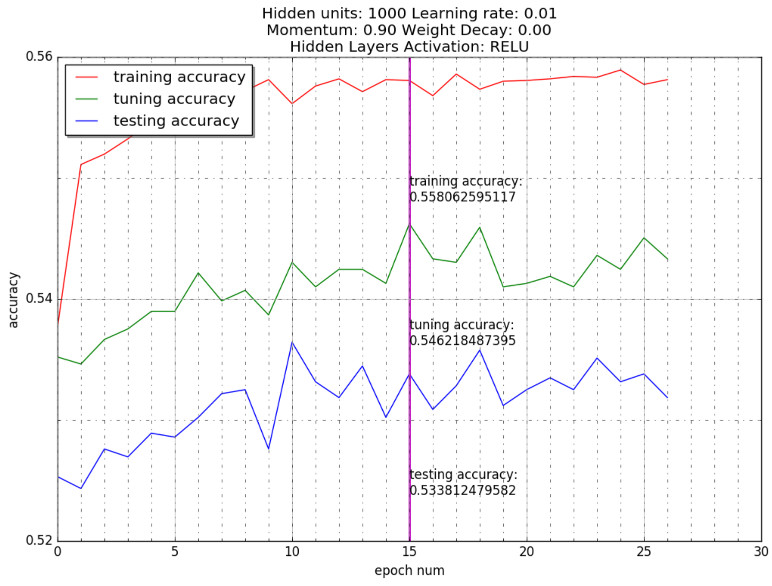

Figure 6: Hidden Units 1000 RELU

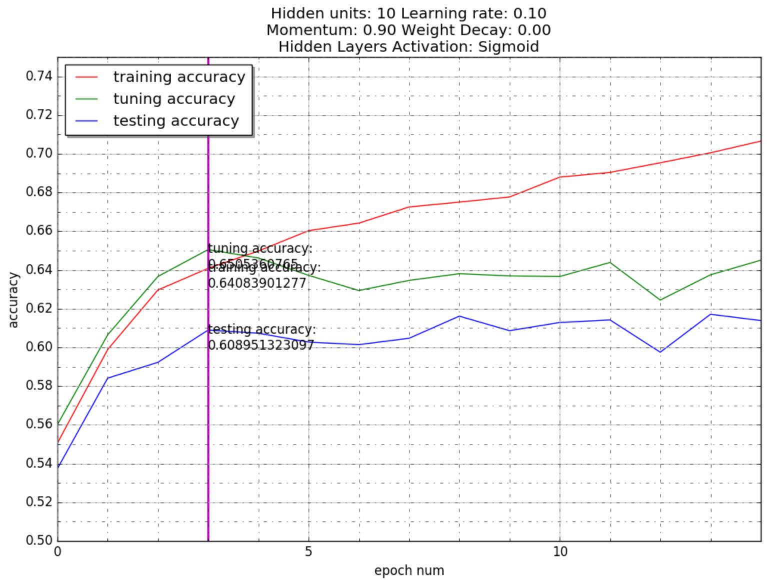

Figure 7: Learning rate 0.1

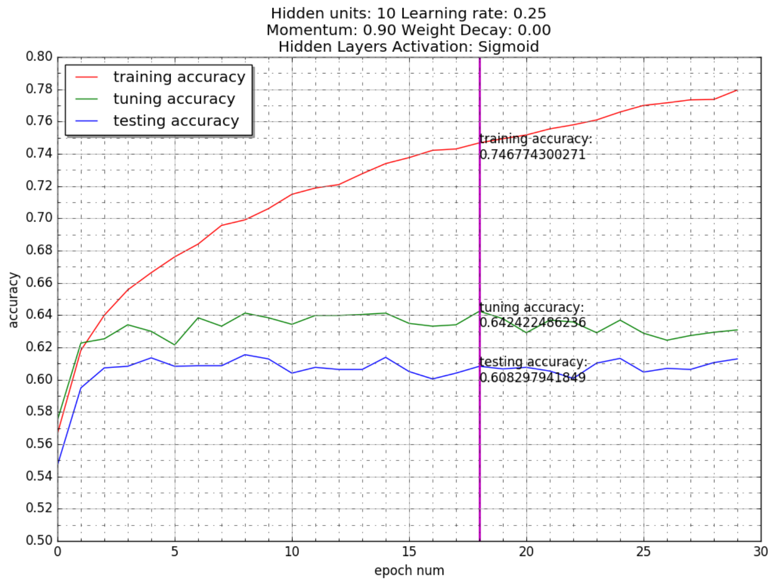

Figure 8: Learning rate 0.25

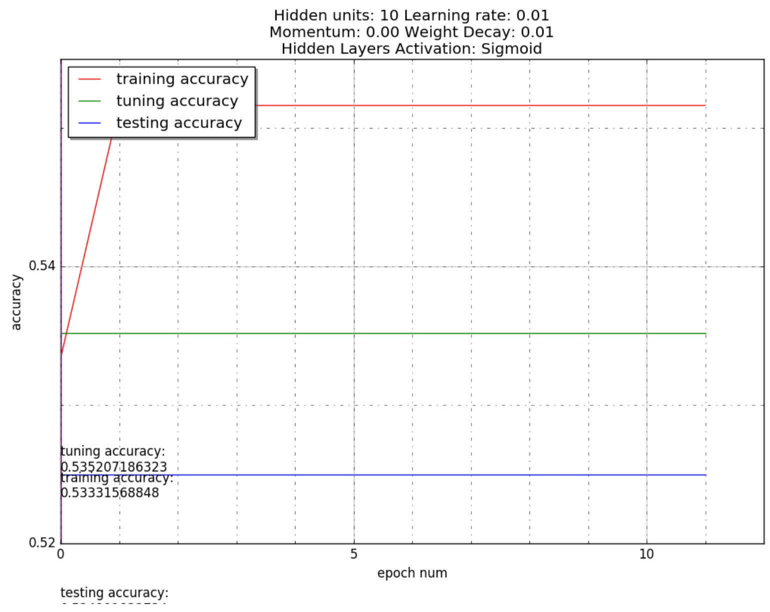

Figure 9: Weight decay 0.01

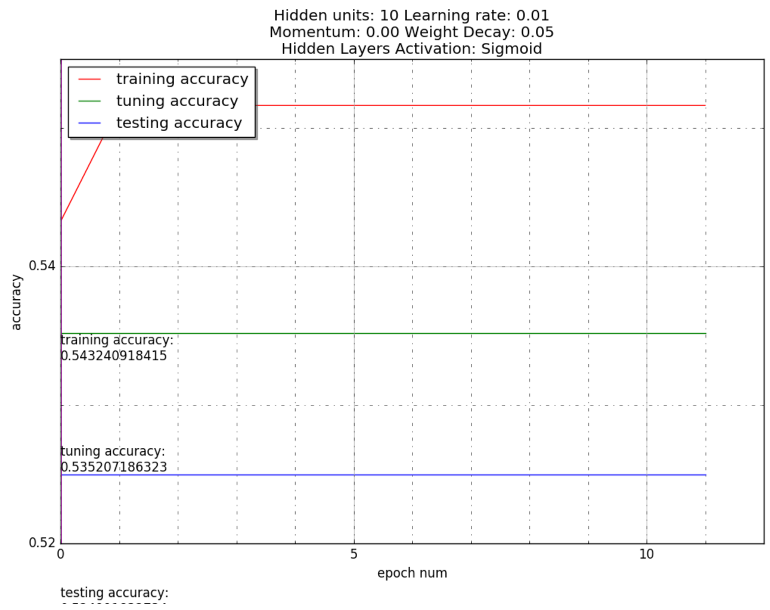

Figure 10: Weight decay 0.05

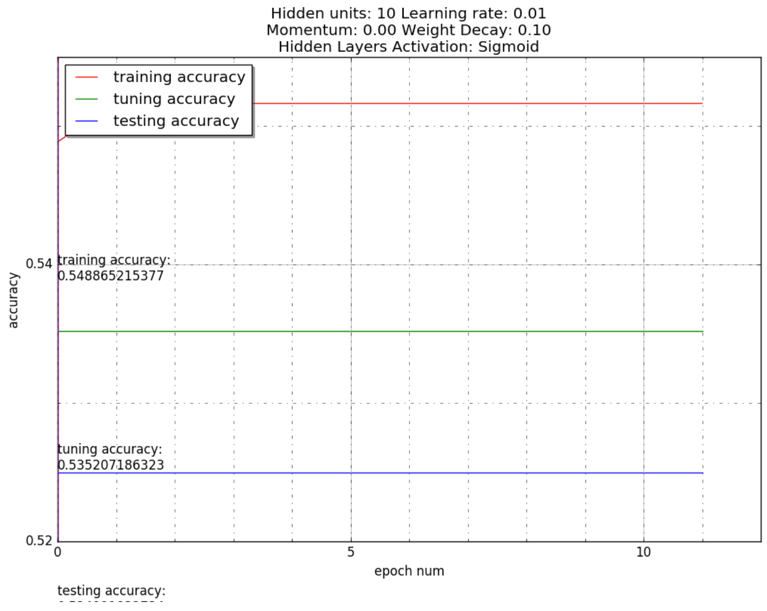

Figure 11: Weight decay 0.10

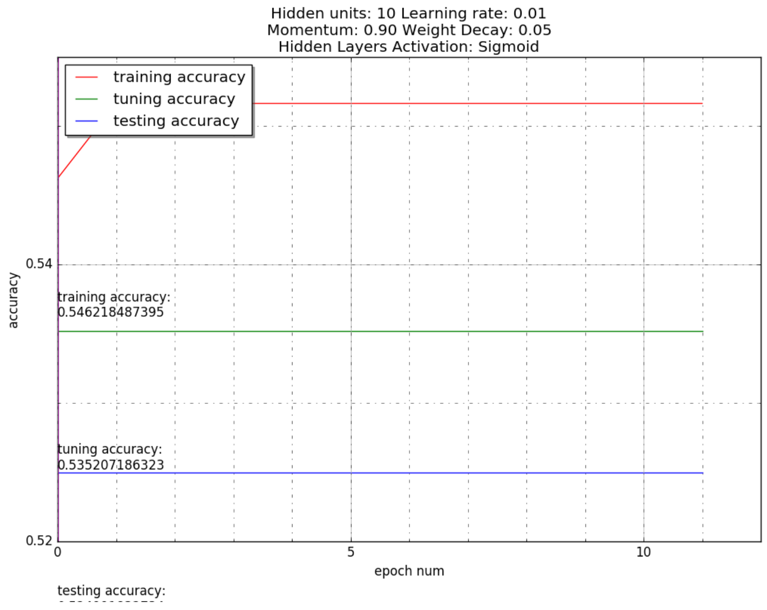

Figure 12: With momentum and weight decay

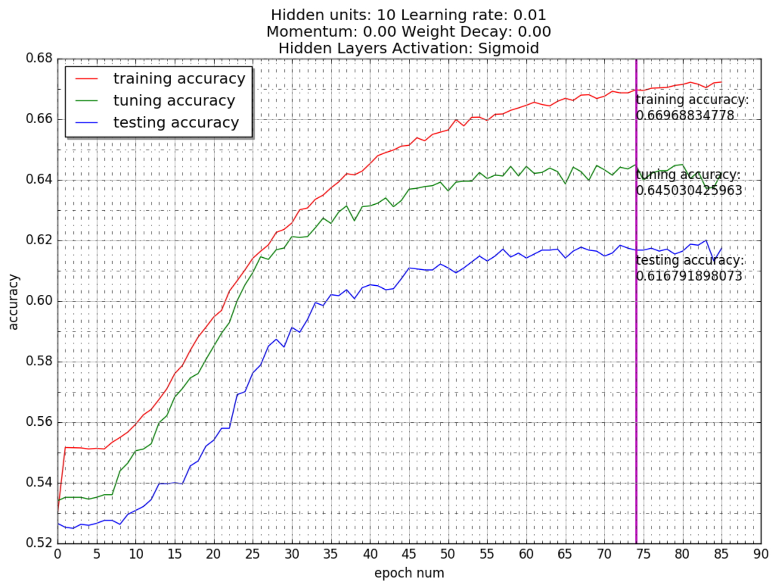

Figure 13: Without momentum and weight decay

2.6 Without momentum and weight decay

Just set learning rate to 0.01. The result is Figure 13.

The accuracy is pretty good comparing to using momentum, but with a slower convergence speed.