Matthew Adler
CS545
Prog1

Experiment 1: Vary number of hidden units.
Experiments with hidden units/nodes: n = 20, 50, and 100.
For each value of n, network is trained on the training set, changing weights after each training example. After each epoch, network's accuracy is calculated on the training set and the test set for plot. Trained network for 50 epochs.

(1)
How does the number of hidden units affect the final accuracy on the test data?

Answer: More hidden units/nodes increase accuracy. This is most noticeable after 20 nodes.

(2)
How does it affect the number of epochs needed for training to converge?

Answer: Less epochs are needed if hidden units are increased.
Since accuracy accelerates faster with more units.

(3)
Is there evidence that any of your networks has overfit to the training data?
If so, what is that evidence?

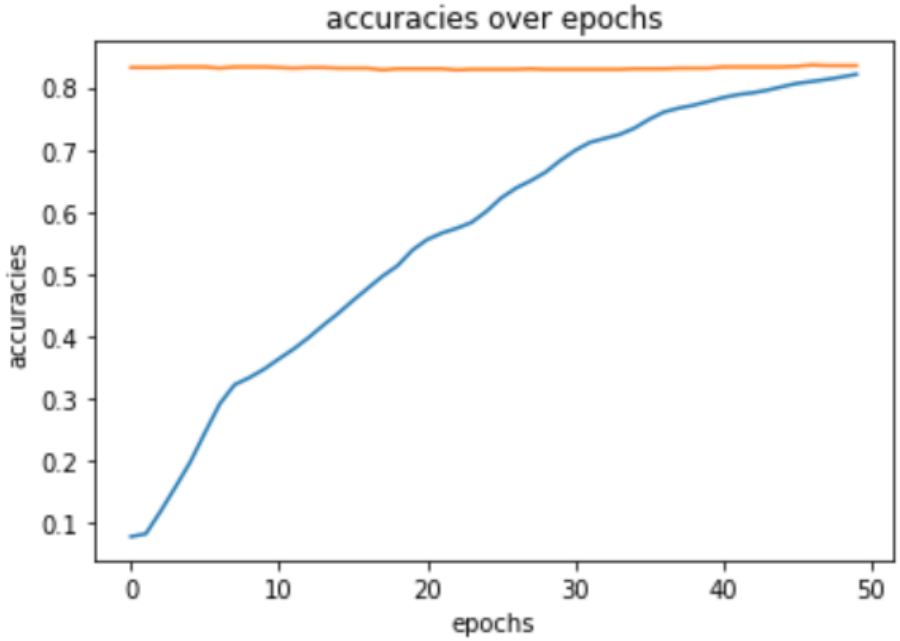
Answer: No overfitting.

(4)
How do your results compare to the results obtained by your perceptron in HW1?

Answer: Accuracy is similar yet it is obtained much faster with hidden layer.

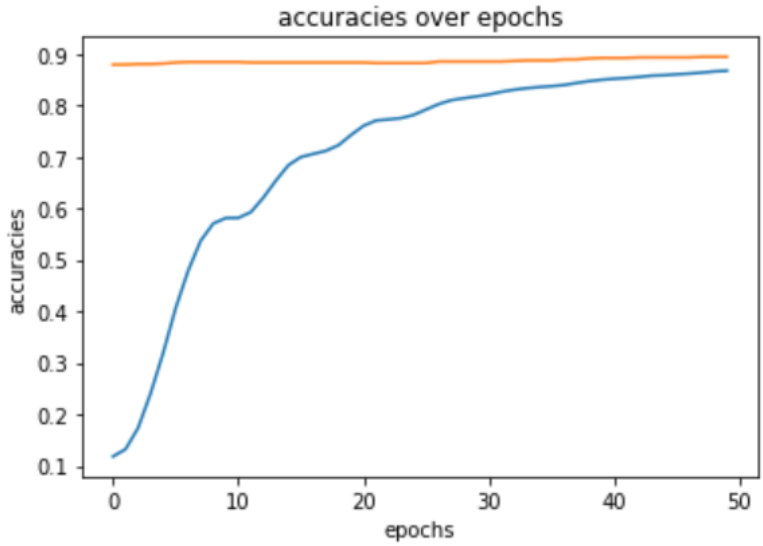
N = 20 nodes:

```
TEST CONFUSION MATRIX
[[ 91   0   1   0   0   3   2   1   0   1]
 [  0 113   3   0   0   2   2   2   4   0]
 [  0   1  75   2   0   1   4   0   6   1]
 [  1   1   0  78   0   6   1   0   3   0]
 [  0   0   1   1  78   3   1   1   2  10]
 [  4   0   1   4   1  60   4   0   5   0]
 [  2   0   1   3   3   4  90   0   0   0]
 [  0   0   3   1   2   4   0  98   1   8]
 [  2   2   1   4   1   2   0   5  85   5]
 [  0   0   0   1   9   3   0   8   4  67]]
```



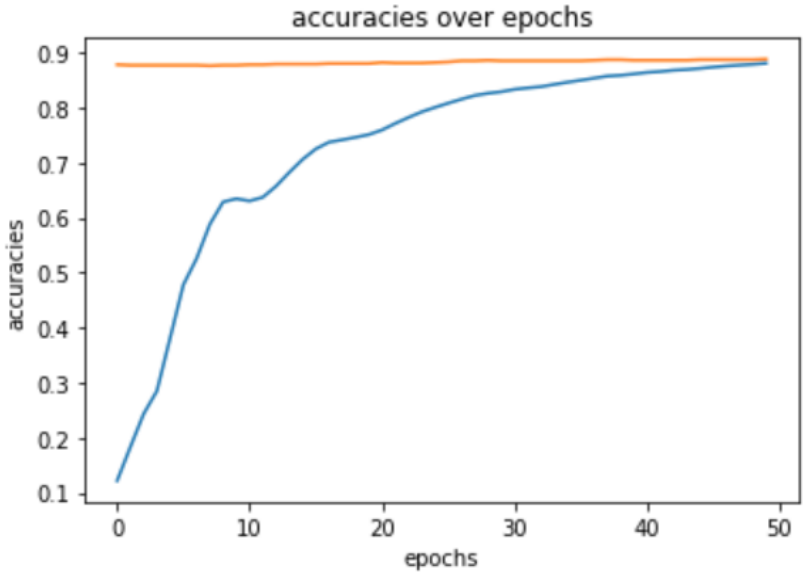
N = 50 nodes:

```
TEST CONFUSION MATRIX
[[ 99   0   1   0   0   1   2   0   1   0]
 [  0 111   1   0   1   1   2   1   2   0]
 [  2   0  80   6   1   1   1   2   2   0]
 [  2   1   1  80   0   3   0   1   2   2]
 [  1   1   1   0  80   1   0   1   0   5]
 [  0   0   1   4   1  75   2   0   4   0]
 [  1   2   5   0   1   0  78   0   0   0]
 [  0   0   3   1   1   1   0 112   0   4]
 [  1   0   1   1   0   4   1   0  78   1]
 [  0   0   0   2   7   2   0   3   2 101]]
```



N = 100 nodes:

```
TEST CONFUSION MATRIX
[[ 92   0   2   1   0   1   0   1   0   0]
 [  0 114   1   0   0   0   0   1   3   0]
 [  1   0  80   2   1   0   2   1   3   0]
 [  0   0   3  94   0   5   0   0   6   1]
 [  1   0   3   0  90   0   3   2   1   8]
 [  3   0   0   5   0  89   3   0   4   0]
 [  0   0   1   1   3   1  94   0   1   1]
 [  0   0   2   1   0   0   0  86   3   4]
 [  1   0   4   2   1   4   0   1  62   1]
 [  0   0   1   2   5   0   0   2   2  87]]
```



Experiment 2: Vary the momentum value.
Here, hidden units is fixed to 100. Momentum values varied during training: 0, 0.25, and 0.5. Train networks using these values as in Experiment 1, and plot the results as in Experiment 1. (Included in plot n=100 and momentum = 0.9 that you completed in

Experiment 1.)

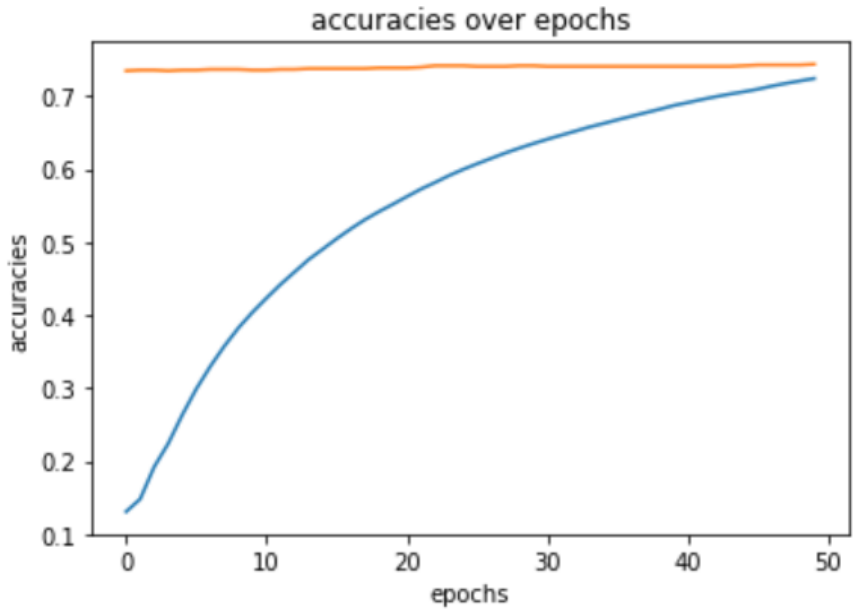
(1)
How does the momentum value affect the final accuracy on the test data?
Answer: More momentum increases final accuracy.

(2)
How does it affect the number of epochs needed for training to converge?
Answer: Momentum accelerates the the accuracy rate. So again, less epochs needed for model to learn.

(3)
Again, is there evidence that any of your networks has overfit to the trainingdata?
If so, what is that evidence?
Answer: No evidence of overfitting because convergence occurs before 50 epochs.

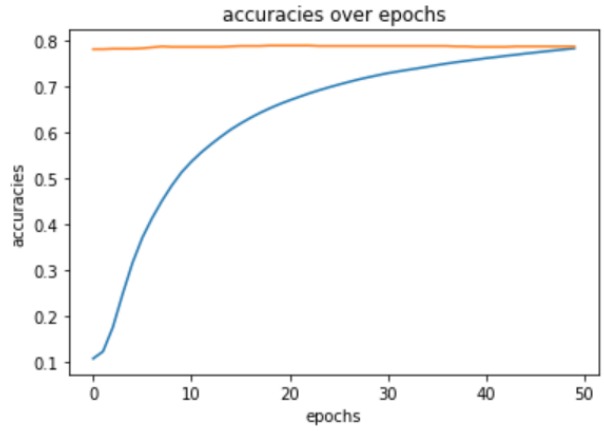
Momentum=0:

```
TEST CONFUSION MATRIX
[[ 85   0   5   1   0   5   1   2   1   3]
 [  0 115   4   4   2   4   1   1   5   0]
 [  1   0  68   8   0   0   7   1  10   2]
 [  1   6   3  74   2   4   0   0   2   0]
 [  1   0   4   0  59   1   2   4   2  11]
 [  9   0   3   4   1  66   3   3   2   1]
 [  5   3   7   2   3   2  76   0   1   0]
 [  0   1   4   1   3   5   2  86   0   6]
 [  0   3   2   7   2   8   1   2  53   8]
 [  0   1   1   1  20   5   0   7   6  62]]
```



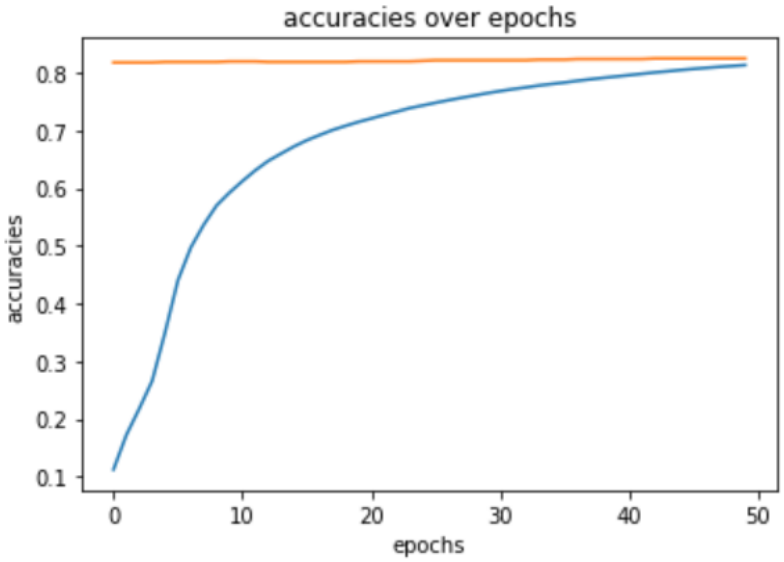
Momentum=.25

```
TEST CONFUSION MATRIX
[[ 94   0   1   0   3   3   4   0   1   0]
 [  0 105   4   3   1   2   1   2   5   1]
 [  3   0  77   3   1   4   2   2   6   1]
 [  1   2   5  60   0  11   2   3   7   2]
 [  0   0   4   0  82   4   1   2   0   8]
 [  4   0   1   4   1  62   4   0   3   1]
 [  3   1   3   0   2   3  79   0   2   2]
 [  0   1   0   1   0   1   0  71   0   8]
 [  2   3   4   7   1   3   1   1  65   3]
 [  0   0   0   2  16   4   0  15   0  93]]
```



Momentum= .5

```
TEST CONFUSION MATRIX
[[111   1   1   2   0   7   0   2   1   3]
 [  0 106   2   0   1   2   1   1   0   1]
 [  2   2  78   2   0   1   6   1   2   0]
 [  2   0   1  73   0   5   0   2   6   1]
 [  0   0   3   0  92   2   0   3   3  10]
 [  4   1   3   4   1  59   1   0   5   0]
 [  5   1   0   2   1   4  80   0   0   1]
 [  1   0   2   2   0   0   0  88   0   8]
 [  2   3   3   4   3   4   0   0  76   5]
 [  0   0   2   1   8   4   0   6   5  62]]
```



Experiment 3: Vary the number of training examples.
In this experiment, the number of
hidden units are set to 100 and momentum 0.9.
Instead of using all of the training examples, two networks are trained,
using respectively one quarter and one half of the training
Plot the results, as in the previous experiments,
plotting accuracy on both the training and test data at the end of each epoch.

(1)
How does the size of the training data affect the final accuracy on the test data?
Answer: Accuracy is higher with larger dataset.

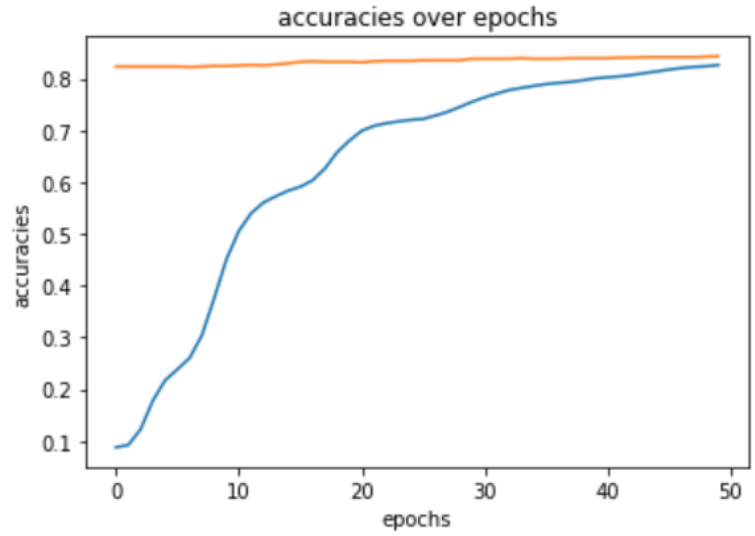
(2)
How does it affect the number of epochs needed for training to converge?

Answer: With a smaller the data set, it takes a longer time to improve accuracy on the model. The larger the datas set it seems that the accuracies increase in a shorter amount of time.

(3)
Again, is there evidence that any of your networks has overfit to the trainingdata?
If so, what is that evidence?
Answer: No overfitting.

¼ Data:

```
TEST CONFUSION MATRIX
[[ 94   0   0   0   0   5   0   0   2   0]
 [  0  99   0   1   1   2   0   2   0   0]
 [  0   0  86   3   3   1   1   1   3   0]
 [  0   1   0  76   0   6   0   4   5   2]
 [  0   0   2   0  69   2   2   3   2   4]
 [  5   1   0  10   0  57   0   1   4   0]
 [  2   0   5   0   1   3  97   0   3   1]
 [  0   1   2   3   4   2   0 104   4   5]
 [  1   2   4   2   1   5   3   0  64   4]
 [  0   0   1   1   6   4   0   6   2  97]]
```



½ Data:

```
TEST CONFUSION MATRIX
[[ 83   0   0   0   0   1   2   0   1   0]
 [  0 110   2   1   1   1   0   0   2   1]
 [  0   2  89   3   0   2   1   2   2   0]
 [  0   1   0  87   0   6   0   0   7   2]
 [  0   0   1   0  92   3   2   1   1   7]
 [  3   1   1   4   1  77   2   1   2   2]
 [  1   0   3   0   3   1  90   0   0   1]
 [  0   1   2   2   0   0   0  99   1   2]
 [  1   1   4   3   0   3   2   0  76   1]
 [  0   0   1   0   3   2   0   4   1  82]]
```