**Neural Network For Digit Recognition**

**Step 1:** Define the architecture of neural network

|  |  |  |  |
| --- | --- | --- | --- |
| No. of input layers: | 1 | No. of input units: | 400 |
| No. of hidden layers: | 1 | No. of hidden units: | 25 |
| No. of output layers: | 1 | No. of class units: | 10 |

(Step 2 – Step 4: Implement these algorithms and functions independently to test if they perform appropriately before we integrate into the training algorithm of our neural network)

**Step 2:** Compute sigmoid gradient function and initialize parameter theta randomly

**Step 3:** Implement feed-forward propagation to compute cost with regularization

**Step 4:** Implement back-propagation with regularization to compute the error matrix and the cumulative partial derivative of cost function with respect to two different parameters theta.

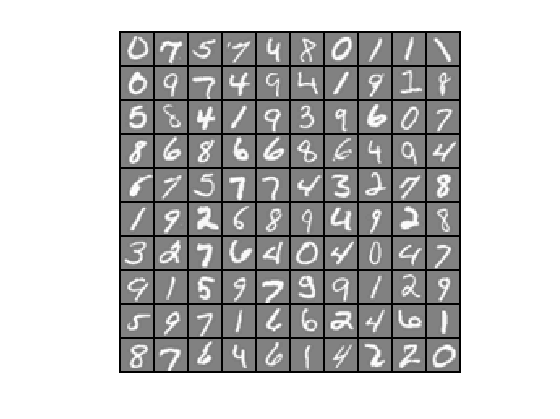
**Step 5:** Train neural network and optimize the convergence of cost value and parameters theta.

**Step 6:** Visualize the representation of hidden units

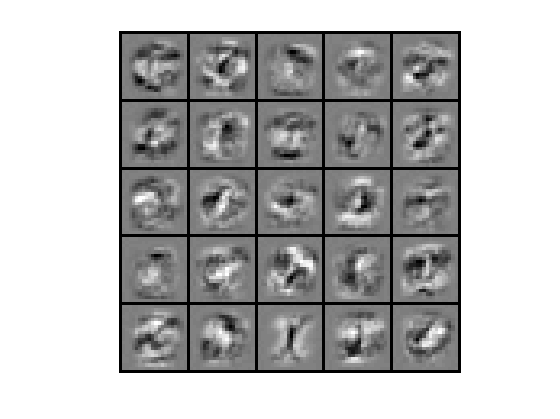
**Step 7:** Display the example image and make predictions based on the input image

**Results:**

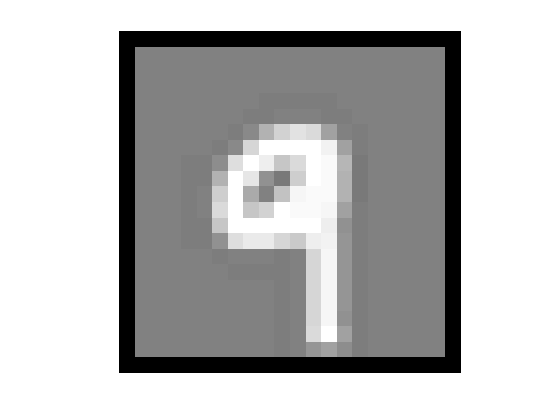
1. Display 100 random digit images from dataset initially.



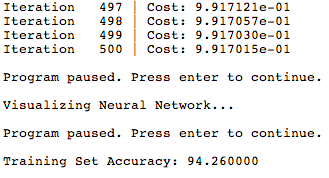
2. Visualization of learning representation about 25 hidden units.



3. One of the predicted images result.



4. Convergence of cost value and training set accuracy.



**Conclusion:**

Since we have a very high accuracy, we’re assured that the current neural network architecture performs pretty well for digit recognition application. This neural network algorithm can be utilized for feature extraction and other purposes. In addition, the accuracy can be improved further by increasing the number of iterations and regularization term.

**Appendix:**

Data resource:

“The MNIST database of handwritten digits” - <http://yann.lecun.com/exdb/mnist/>