Multi-class Classification and Neural Network

Table of Contents

1. Multi-class Classification

1.1 Dataset

There are 5000 training examples in MNIST`.mat`, where each training example is a 20 pixel by 20 pixel grayscale image of the digit. Each pixel is represented by a floating point number indicating the grayscale intensity at that location. The 20 by 20 grid of pixels is unrolled into a 400-dimensional vector.

This gives us a 5000 by 400 matrix X where every row is a training example for a handwritten digit image.

$$X = \begin{bmatrix} -(x^{(1)})^T - \\ -(x^{(2)})^T - \\ \vdots \\ -(x^{(m)})^T - \end{bmatrix}$$

The second part of the training set is a 5000-dimensional vector y that contains labels for the training set. To make things more compatible with MATLAB, a '0' digit is labeled as '10', while the digits '1' to '9' are labeled as '1' to '9' in their natural order.

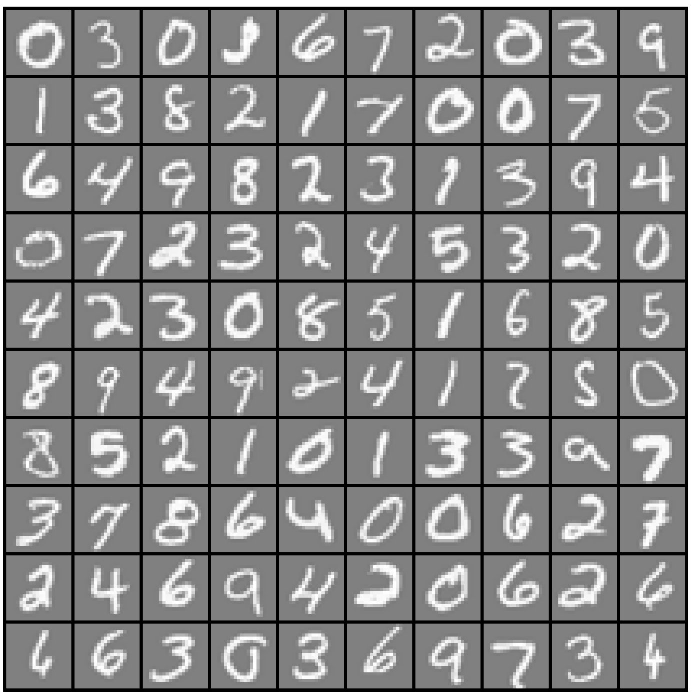
```
clear

load('MNIST.mat'); % Load MNIST file
data = [X y];
data = data(randperm(size(data, 1)), :); % Shuffle the data set

% Split the data set : 80% training set, 20% test set
```

```
TrainX = data(1:4000,1:400);
Trainy = data(1:4000,401);
TestX = data(4001:end,1:400);
Testy = data(4001:end,401);

fprintf('\nNumber of training examples is : %d & number of features is %d\n', size(Tra
```

```
Number of training examples is : 4000 & number of features is 400
```

1.2 Visualizing the data

```
Trainm = size(TrainX, 1);

% Randomly select 100 data points to display
rand_indices = randperm(Trainm); % Random permutation
select = TrainX(rand_indices(1:100), :);
displayData(select);
```



1.3 One-vs-all classication

We use one-vs-all logistic regression models to build a multi-class classifier. Since there are 10 classes, we need to train 10 separate logistic regression classifiers.

```
num_labels = 10; % 10 labels, from 1 to 10
lambda = 0.1;
[all_theta] = oneVsAll(TrainX, Trainy, num_labels, lambda);
```

```
Iteration     1 | Cost: 2.768070e-01
Iteration     2 | Cost: 8.130227e-02
Iteration     3 | Cost: 5.581083e-02
Iteration     4 | Cost: 4.461499e-02
Iteration     5 | Cost: 4.184156e-02
Iteration     6 | Cost: 3.312331e-02
Iteration     7 | Cost: 3.004431e-02
Iteration     8 | Cost: 2.933813e-02
Iteration     9 | Cost: 2.639368e-02
Iteration    10 | Cost: 2.574111e-02
Iteration    11 | Cost: 2.443374e-02
Iteration    12 | Cost: 2.386844e-02
Iteration    13 | Cost: 2.309251e-02
Iteration    14 | Cost: 2.256434e-02
Iteration    15 | Cost: 2.188966e-02
Iteration    16 | Cost: 2.173841e-02
Iteration    17 | Cost: 2.151643e-02
Iteration    18 | Cost: 2.091207e-02
Iteration    19 | Cost: 2.079277e-02
Iteration    20 | Cost: 2.047197e-02
Iteration    21 | Cost: 1.946712e-02
Iteration    22 | Cost: 1.889963e-02
Iteration    23 | Cost: 1.847389e-02
Iteration    24 | Cost: 1.829850e-02
Iteration    25 | Cost: 1.805435e-02
Iteration    26 | Cost: 1.700980e-02
Iteration    27 | Cost: 1.612403e-02
Iteration    28 | Cost: 1.588481e-02
Iteration    29 | Cost: 1.562374e-02
Iteration    30 | Cost: 1.503079e-02
Iteration    31 | Cost: 1.498270e-02
Iteration    32 | Cost: 1.486124e-02
Iteration    33 | Cost: 1.482365e-02
Iteration    34 | Cost: 1.453539e-02
Iteration    35 | Cost: 1.441521e-02
Iteration    36 | Cost: 1.427937e-02
Iteration    37 | Cost: 1.424067e-02
Iteration    38 | Cost: 1.419304e-02
Iteration    39 | Cost: 1.414759e-02
Iteration    40 | Cost: 1.412906e-02
Iteration    41 | Cost: 1.404611e-02
Iteration    42 | Cost: 1.400631e-02
Iteration    43 | Cost: 1.399004e-02
Iteration    44 | Cost: 1.387910e-02
Iteration    45 | Cost: 1.375246e-02
Iteration    46 | Cost: 1.331300e-02
Iteration    47 | Cost: 1.321251e-02
Iteration    48 | Cost: 1.312909e-02
Iteration    49 | Cost: 1.300368e-02
Iteration    50 | Cost: 1.294146e-02
Iteration     1 | Cost: 3.433147e-01
Iteration     2 | Cost: 3.100673e-01
Iteration     3 | Cost: 1.791751e-01
Iteration     4 | Cost: 1.710919e-01
Iteration     5 | Cost: 1.470413e-01
Iteration     6 | Cost: 1.210380e-01
Iteration     7 | Cost: 1.112865e-01
Iteration     8 | Cost: 1.083943e-01
Iteration     9 | Cost: 9.540719e-02
Iteration    10 | Cost: 9.411567e-02
Iteration    11 | Cost: 8.956868e-02
Iteration    12 | Cost: 8.737726e-02
Iteration    13 | Cost: 8.479390e-02
Iteration    14 | Cost: 8.049757e-02
```

```
Iteration    15 | Cost: 7.457626e-02
Iteration    16 | Cost: 7.307339e-02
Iteration    17 | Cost: 7.133888e-02
Iteration    18 | Cost: 7.106699e-02
Iteration    19 | Cost: 7.050432e-02
Iteration    20 | Cost: 6.898150e-02
Iteration    21 | Cost: 6.807308e-02
Iteration    22 | Cost: 6.697176e-02
Iteration    23 | Cost: 6.626507e-02
Iteration    24 | Cost: 6.595551e-02
Iteration    25 | Cost: 6.577446e-02
Iteration    26 | Cost: 6.568032e-02
Iteration    27 | Cost: 6.465156e-02
Iteration    28 | Cost: 6.402474e-02
Iteration    29 | Cost: 6.351679e-02
Iteration    30 | Cost: 6.258386e-02
Iteration    31 | Cost: 6.079200e-02
Iteration    32 | Cost: 6.070269e-02
Iteration    33 | Cost: 6.033647e-02
Iteration    34 | Cost: 6.001955e-02
Iteration    35 | Cost: 5.994376e-02
Iteration    36 | Cost: 5.961728e-02
Iteration    37 | Cost: 5.947056e-02
Iteration    38 | Cost: 5.904313e-02
Iteration    39 | Cost: 5.881884e-02
Iteration    40 | Cost: 5.847783e-02
Iteration    41 | Cost: 5.842427e-02
Iteration    42 | Cost: 5.828454e-02
Iteration    43 | Cost: 5.822813e-02
Iteration    44 | Cost: 5.807656e-02
Iteration    45 | Cost: 5.798748e-02
Iteration    46 | Cost: 5.778228e-02
Iteration    47 | Cost: 5.773282e-02
Iteration    48 | Cost: 5.758423e-02
Iteration    49 | Cost: 5.753731e-02
Iteration    50 | Cost: 5.707422e-02
Iteration     1 | Cost: 3.400312e-01
Iteration     2 | Cost: 2.161711e-01
Iteration     3 | Cost: 1.919540e-01
Iteration     4 | Cost: 1.825033e-01
Iteration     5 | Cost: 1.474602e-01
Iteration     6 | Cost: 1.190199e-01
Iteration     7 | Cost: 1.125949e-01
Iteration     8 | Cost: 1.081076e-01
Iteration     9 | Cost: 1.068231e-01
Iteration    10 | Cost: 1.054268e-01
Iteration    11 | Cost: 1.019297e-01
Iteration    12 | Cost: 1.007221e-01
Iteration    13 | Cost: 9.194052e-02
Iteration    14 | Cost: 9.105077e-02
Iteration    15 | Cost: 8.302017e-02
Iteration    16 | Cost: 8.228255e-02
Iteration    17 | Cost: 8.005345e-02
Iteration    18 | Cost: 7.910822e-02
Iteration    19 | Cost: 7.793502e-02
Iteration    20 | Cost: 7.777222e-02
Iteration    21 | Cost: 7.616886e-02
Iteration    22 | Cost: 7.518654e-02
Iteration    23 | Cost: 7.395680e-02
Iteration    24 | Cost: 7.383634e-02
Iteration    25 | Cost: 7.302409e-02
Iteration    26 | Cost: 7.212557e-02
Iteration    27 | Cost: 7.132024e-02
Iteration    28 | Cost: 7.115528e-02
```

```
Iteration    29 | Cost: 7.053449e-02
Iteration    30 | Cost: 7.040005e-02
Iteration    31 | Cost: 6.989788e-02
Iteration    32 | Cost: 6.981716e-02
Iteration    33 | Cost: 6.960381e-02
Iteration    34 | Cost: 6.928838e-02
Iteration    35 | Cost: 6.920779e-02
Iteration    36 | Cost: 6.903721e-02
Iteration    37 | Cost: 6.827322e-02
Iteration    38 | Cost: 6.821849e-02
Iteration    39 | Cost: 6.799269e-02
Iteration    40 | Cost: 6.795781e-02
Iteration    41 | Cost: 6.781234e-02
Iteration    42 | Cost: 6.778038e-02
Iteration    43 | Cost: 6.757779e-02
Iteration    44 | Cost: 6.751337e-02
Iteration    45 | Cost: 6.706686e-02
Iteration    46 | Cost: 6.699385e-02
Iteration    47 | Cost: 6.493038e-02
Iteration    48 | Cost: 6.462804e-02
Iteration    49 | Cost: 6.350839e-02
Iteration    50 | Cost: 6.348809e-02
Iteration     1 | Cost: 3.203858e-01
Iteration     2 | Cost: 3.006680e-01
Iteration     3 | Cost: 1.661189e-01
Iteration     4 | Cost: 1.433660e-01
Iteration     5 | Cost: 1.043891e-01
Iteration     6 | Cost: 9.280396e-02
Iteration     7 | Cost: 8.428500e-02
Iteration     8 | Cost: 7.760851e-02
Iteration     9 | Cost: 6.880932e-02
Iteration    10 | Cost: 6.312920e-02
Iteration    11 | Cost: 6.276326e-02
Iteration    12 | Cost: 6.046786e-02
Iteration    13 | Cost: 5.699720e-02
Iteration    14 | Cost: 5.683354e-02
Iteration    15 | Cost: 5.514956e-02
Iteration    16 | Cost: 5.428097e-02
Iteration    17 | Cost: 5.427030e-02
Iteration    18 | Cost: 5.358364e-02
Iteration    19 | Cost: 5.209144e-02
Iteration    20 | Cost: 5.196679e-02
Iteration    21 | Cost: 5.090708e-02
Iteration    22 | Cost: 5.042389e-02
Iteration    23 | Cost: 4.860446e-02
Iteration    24 | Cost: 4.845615e-02
Iteration    25 | Cost: 4.793735e-02
Iteration    26 | Cost: 4.640534e-02
Iteration    27 | Cost: 4.572783e-02
Iteration    28 | Cost: 4.558786e-02
Iteration    29 | Cost: 4.472370e-02
Iteration    30 | Cost: 4.346276e-02
Iteration    31 | Cost: 4.337731e-02
Iteration    32 | Cost: 4.231630e-02
Iteration    33 | Cost: 4.216756e-02
Iteration    34 | Cost: 4.200682e-02
Iteration    35 | Cost: 4.188329e-02
Iteration    36 | Cost: 4.179766e-02
Iteration    37 | Cost: 4.166585e-02
Iteration    38 | Cost: 3.997629e-02
Iteration    39 | Cost: 3.977579e-02
Iteration    40 | Cost: 3.920292e-02
Iteration    41 | Cost: 3.889167e-02
Iteration    42 | Cost: 3.811906e-02
```

```
Iteration    43 | Cost: 3.803260e-02
Iteration    44 | Cost: 3.722797e-02
Iteration    45 | Cost: 3.714837e-02
Iteration    46 | Cost: 3.661180e-02
Iteration    47 | Cost: 3.638670e-02
Iteration    48 | Cost: 3.624654e-02
Iteration    49 | Cost: 3.623843e-02
Iteration    50 | Cost: 3.615365e-02
Iteration     1 | Cost: 3.372813e-01
Iteration     2 | Cost: 2.271945e-01
Iteration     3 | Cost: 1.904706e-01
Iteration     4 | Cost: 1.394359e-01
Iteration     5 | Cost: 1.352487e-01
Iteration     6 | Cost: 1.179584e-01
Iteration     7 | Cost: 1.173401e-01
Iteration     8 | Cost: 1.098052e-01
Iteration     9 | Cost: 1.071514e-01
Iteration    10 | Cost: 1.037833e-01
Iteration    11 | Cost: 1.013342e-01
Iteration    12 | Cost: 9.836739e-02
Iteration    13 | Cost: 9.603993e-02
Iteration    14 | Cost: 9.221647e-02
Iteration    15 | Cost: 9.125504e-02
Iteration    16 | Cost: 8.825540e-02
Iteration    17 | Cost: 8.789739e-02
Iteration    18 | Cost: 8.646616e-02
Iteration    19 | Cost: 8.599072e-02
Iteration    20 | Cost: 8.495041e-02
Iteration    21 | Cost: 8.471359e-02
Iteration    22 | Cost: 8.148771e-02
Iteration    23 | Cost: 8.127343e-02
Iteration    24 | Cost: 8.053452e-02
Iteration    25 | Cost: 8.038567e-02
Iteration    26 | Cost: 7.919835e-02
Iteration    27 | Cost: 7.883334e-02
Iteration    28 | Cost: 7.661190e-02
Iteration    29 | Cost: 7.632128e-02
Iteration    30 | Cost: 7.284697e-02
Iteration    31 | Cost: 7.200045e-02
Iteration    32 | Cost: 7.083804e-02
Iteration    33 | Cost: 7.073059e-02
Iteration    34 | Cost: 7.043664e-02
Iteration    35 | Cost: 6.974063e-02
Iteration    36 | Cost: 6.950092e-02
Iteration    37 | Cost: 6.933886e-02
Iteration    38 | Cost: 6.918895e-02
Iteration    39 | Cost: 6.897637e-02
Iteration    40 | Cost: 6.831349e-02
Iteration    41 | Cost: 6.781918e-02
Iteration    42 | Cost: 6.746691e-02
Iteration    43 | Cost: 6.739265e-02
Iteration    44 | Cost: 6.718573e-02
Iteration    45 | Cost: 6.700467e-02
Iteration    46 | Cost: 6.696752e-02
Iteration    47 | Cost: 6.653702e-02
Iteration    48 | Cost: 6.646542e-02
Iteration    49 | Cost: 6.629869e-02
Iteration    50 | Cost: 6.593066e-02
Iteration     1 | Cost: 3.376520e-01
Iteration     2 | Cost: 2.248383e-01
Iteration     3 | Cost: 9.614777e-02
Iteration     4 | Cost: 8.719324e-02
Iteration     5 | Cost: 6.563953e-02
Iteration     6 | Cost: 6.181944e-02
```

```
Iteration     7 | Cost: 4.766731e-02
Iteration     8 | Cost: 4.599773e-02
Iteration     9 | Cost: 4.375595e-02
Iteration    10 | Cost: 4.161750e-02
Iteration    11 | Cost: 4.104711e-02
Iteration    12 | Cost: 3.905780e-02
Iteration    13 | Cost: 3.833761e-02
Iteration    14 | Cost: 3.635672e-02
Iteration    15 | Cost: 3.612785e-02
Iteration    16 | Cost: 3.392163e-02
Iteration    17 | Cost: 3.368699e-02
Iteration    18 | Cost: 3.277080e-02
Iteration    19 | Cost: 3.274950e-02
Iteration    20 | Cost: 3.253543e-02
Iteration    21 | Cost: 3.226342e-02
Iteration    22 | Cost: 3.216965e-02
Iteration    23 | Cost: 3.206070e-02
Iteration    24 | Cost: 3.167041e-02
Iteration    25 | Cost: 3.124999e-02
Iteration    26 | Cost: 2.887603e-02
Iteration    27 | Cost: 2.801260e-02
Iteration    28 | Cost: 2.721163e-02
Iteration    29 | Cost: 2.682039e-02
Iteration    30 | Cost: 2.672353e-02
Iteration    31 | Cost: 2.657665e-02
Iteration    32 | Cost: 2.647602e-02
Iteration    33 | Cost: 2.634837e-02
Iteration    34 | Cost: 2.630896e-02
Iteration    35 | Cost: 2.616855e-02
Iteration    36 | Cost: 2.590327e-02
Iteration    37 | Cost: 2.565429e-02
Iteration    38 | Cost: 2.544277e-02
Iteration    39 | Cost: 2.524832e-02
Iteration    40 | Cost: 2.517930e-02
Iteration    41 | Cost: 2.509684e-02
Iteration    42 | Cost: 2.507910e-02
Iteration    43 | Cost: 2.495798e-02
Iteration    44 | Cost: 2.488763e-02
Iteration    45 | Cost: 2.463362e-02
Iteration    46 | Cost: 2.450758e-02
Iteration    47 | Cost: 2.437343e-02
Iteration    48 | Cost: 2.430692e-02
Iteration    49 | Cost: 2.425803e-02
Iteration    50 | Cost: 2.414479e-02
Iteration     1 | Cost: 3.134921e-01
Iteration     2 | Cost: 1.933074e-01
Iteration     3 | Cost: 1.065004e-01
Iteration     4 | Cost: 9.904031e-02
Iteration     5 | Cost: 9.386919e-02
Iteration     6 | Cost: 7.116096e-02
Iteration     7 | Cost: 6.726114e-02
Iteration     8 | Cost: 6.577063e-02
Iteration     9 | Cost: 6.375026e-02
Iteration    10 | Cost: 6.171458e-02
Iteration    11 | Cost: 5.869972e-02
Iteration    12 | Cost: 5.720547e-02
Iteration    13 | Cost: 5.655362e-02
Iteration    14 | Cost: 5.634876e-02
Iteration    15 | Cost: 5.461460e-02
Iteration    16 | Cost: 5.352073e-02
Iteration    17 | Cost: 5.307681e-02
Iteration    18 | Cost: 5.125072e-02
Iteration    19 | Cost: 5.112611e-02
Iteration    20 | Cost: 5.082164e-02
```

```
Iteration    21 | Cost: 5.074880e-02
Iteration    22 | Cost: 5.043220e-02
Iteration    23 | Cost: 4.949250e-02
Iteration    24 | Cost: 4.886002e-02
Iteration    25 | Cost: 4.841607e-02
Iteration    26 | Cost: 4.758265e-02
Iteration    27 | Cost: 4.684269e-02
Iteration    28 | Cost: 4.641914e-02
Iteration    29 | Cost: 4.634745e-02
Iteration    30 | Cost: 4.620580e-02
Iteration    31 | Cost: 4.558816e-02
Iteration    32 | Cost: 4.507155e-02
Iteration    33 | Cost: 4.464628e-02
Iteration    34 | Cost: 4.406214e-02
Iteration    35 | Cost: 4.350174e-02
Iteration    36 | Cost: 4.234435e-02
Iteration    37 | Cost: 4.209072e-02
Iteration    38 | Cost: 4.182942e-02
Iteration    39 | Cost: 4.155141e-02
Iteration    40 | Cost: 4.146916e-02
Iteration    41 | Cost: 4.138037e-02
Iteration    42 | Cost: 4.118964e-02
Iteration    43 | Cost: 4.106363e-02
Iteration    44 | Cost: 4.053326e-02
Iteration    45 | Cost: 4.044278e-02
Iteration    46 | Cost: 4.031445e-02
Iteration    47 | Cost: 3.991882e-02
Iteration    48 | Cost: 3.895439e-02
Iteration    49 | Cost: 3.846367e-02
Iteration    50 | Cost: 3.671225e-02
Iteration     1 | Cost: 3.722350e-01
Iteration     2 | Cost: 2.577686e-01
Iteration     3 | Cost: 2.317431e-01
Iteration     4 | Cost: 2.277396e-01
Iteration     5 | Cost: 2.095518e-01
Iteration     6 | Cost: 1.602041e-01
Iteration     7 | Cost: 1.563103e-01
Iteration     8 | Cost: 1.438199e-01
Iteration     9 | Cost: 1.387087e-01
Iteration    10 | Cost: 1.280233e-01
Iteration    11 | Cost: 1.232502e-01
Iteration    12 | Cost: 1.115149e-01
Iteration    13 | Cost: 1.099548e-01
Iteration    14 | Cost: 1.083532e-01
Iteration    15 | Cost: 1.060030e-01
Iteration    16 | Cost: 1.025436e-01
Iteration    17 | Cost: 9.902829e-02
Iteration    18 | Cost: 9.867317e-02
Iteration    19 | Cost: 9.788245e-02
Iteration    20 | Cost: 9.670711e-02
Iteration    21 | Cost: 9.556376e-02
Iteration    22 | Cost: 9.325506e-02
Iteration    23 | Cost: 9.237397e-02
Iteration    24 | Cost: 9.158591e-02
Iteration    25 | Cost: 9.043734e-02
Iteration    26 | Cost: 9.022764e-02
Iteration    27 | Cost: 8.932259e-02
Iteration    28 | Cost: 8.924360e-02
Iteration    29 | Cost: 8.873451e-02
Iteration    30 | Cost: 8.733989e-02
Iteration    31 | Cost: 8.579115e-02
Iteration    32 | Cost: 8.532365e-02
Iteration    33 | Cost: 8.370881e-02
Iteration    34 | Cost: 8.333801e-02
```

```
Iteration    35 | Cost: 8.178925e-02
Iteration    36 | Cost: 8.164617e-02
Iteration    37 | Cost: 8.105588e-02
Iteration    38 | Cost: 8.088943e-02
Iteration    39 | Cost: 8.027531e-02
Iteration    40 | Cost: 8.008884e-02
Iteration    41 | Cost: 7.991497e-02
Iteration    42 | Cost: 7.987546e-02
Iteration    43 | Cost: 7.943919e-02
Iteration    44 | Cost: 7.937507e-02
Iteration    45 | Cost: 7.916172e-02
Iteration    46 | Cost: 7.837445e-02
Iteration    47 | Cost: 7.837085e-02
Iteration    48 | Cost: 7.829168e-02
Iteration    49 | Cost: 7.793581e-02
Iteration    50 | Cost: 7.790649e-02
Iteration     1 | Cost: 3.391385e-01
Iteration     2 | Cost: 2.521806e-01
Iteration     3 | Cost: 2.254324e-01
Iteration     4 | Cost: 2.136981e-01
Iteration     5 | Cost: 1.915254e-01
Iteration     6 | Cost: 1.742694e-01
Iteration     7 | Cost: 1.539674e-01
Iteration     8 | Cost: 1.383133e-01
Iteration     9 | Cost: 1.295859e-01
Iteration    10 | Cost: 1.240513e-01
Iteration    11 | Cost: 1.174094e-01
Iteration    12 | Cost: 1.124735e-01
Iteration    13 | Cost: 1.109834e-01
Iteration    14 | Cost: 1.079516e-01
Iteration    15 | Cost: 1.072126e-01
Iteration    16 | Cost: 1.043164e-01
Iteration    17 | Cost: 1.012420e-01
Iteration    18 | Cost: 1.003501e-01
Iteration    19 | Cost: 9.850245e-02
Iteration    20 | Cost: 9.607982e-02
Iteration    21 | Cost: 9.561822e-02
Iteration    22 | Cost: 9.515561e-02
Iteration    23 | Cost: 9.479832e-02
Iteration    24 | Cost: 9.474759e-02
Iteration    25 | Cost: 9.410746e-02
Iteration    26 | Cost: 9.304556e-02
Iteration    27 | Cost: 9.176875e-02
Iteration    28 | Cost: 8.834092e-02
Iteration    29 | Cost: 8.447311e-02
Iteration    30 | Cost: 8.366989e-02
Iteration    31 | Cost: 8.267476e-02
Iteration    32 | Cost: 8.183278e-02
Iteration    33 | Cost: 7.967708e-02
Iteration    34 | Cost: 7.948874e-02
Iteration    35 | Cost: 7.915657e-02
Iteration    36 | Cost: 7.903490e-02
Iteration    37 | Cost: 7.851138e-02
Iteration    38 | Cost: 7.835326e-02
Iteration    39 | Cost: 7.783476e-02
Iteration    40 | Cost: 7.722381e-02
Iteration    41 | Cost: 7.719839e-02
Iteration    42 | Cost: 7.703820e-02
Iteration    43 | Cost: 7.694133e-02
Iteration    44 | Cost: 7.690969e-02
Iteration    45 | Cost: 7.679256e-02
Iteration    46 | Cost: 7.651871e-02
Iteration    47 | Cost: 7.646843e-02
Iteration    48 | Cost: 7.613331e-02
```

```
Iteration    49 | Cost: 7.604504e-02
Iteration    50 | Cost: 7.491297e-02
Iteration     1 | Cost: 3.481387e-01
Iteration     2 | Cost: 2.096530e-01
Iteration     3 | Cost: 1.122660e-01
Iteration     4 | Cost: 1.000092e-01
Iteration     5 | Cost: 5.194748e-02
Iteration     6 | Cost: 4.908835e-02
Iteration     7 | Cost: 3.656853e-02
Iteration     8 | Cost: 3.587326e-02
Iteration     9 | Cost: 2.734057e-02
Iteration    10 | Cost: 2.627651e-02
Iteration    11 | Cost: 2.408440e-02
Iteration    12 | Cost: 2.318731e-02
Iteration    13 | Cost: 2.290613e-02
Iteration    14 | Cost: 2.254174e-02
Iteration    15 | Cost: 2.160925e-02
Iteration    16 | Cost: 2.030736e-02
Iteration    17 | Cost: 2.026256e-02
Iteration    18 | Cost: 1.924678e-02
Iteration    19 | Cost: 1.864706e-02
Iteration    20 | Cost: 1.674644e-02
Iteration    21 | Cost: 1.651062e-02
Iteration    22 | Cost: 1.461474e-02
Iteration    23 | Cost: 1.432945e-02
Iteration    24 | Cost: 1.348233e-02
Iteration    25 | Cost: 1.338917e-02
Iteration    26 | Cost: 1.305911e-02
Iteration    27 | Cost: 1.302293e-02
Iteration    28 | Cost: 1.286920e-02
Iteration    29 | Cost: 1.283406e-02
Iteration    30 | Cost: 1.271397e-02
Iteration    31 | Cost: 1.267696e-02
Iteration    32 | Cost: 1.232009e-02
Iteration    33 | Cost: 1.218272e-02
Iteration    34 | Cost: 1.163568e-02
Iteration    35 | Cost: 1.159242e-02
Iteration    36 | Cost: 1.146207e-02
Iteration    37 | Cost: 1.130846e-02
Iteration    38 | Cost: 1.115364e-02
Iteration    39 | Cost: 1.111483e-02
Iteration    40 | Cost: 1.105980e-02
Iteration    41 | Cost: 1.104248e-02
Iteration    42 | Cost: 1.098226e-02
Iteration    43 | Cost: 1.097055e-02
Iteration    44 | Cost: 1.092078e-02
Iteration    45 | Cost: 1.091438e-02
Iteration    46 | Cost: 1.083225e-02
Iteration    47 | Cost: 1.079000e-02
Iteration    48 | Cost: 1.077640e-02
Iteration    49 | Cost: 1.060389e-02
Iteration    50 | Cost: 1.050699e-02
```

1.4 One-vs-all prediction

After training our one-vs-all classifier, we can now use it to predict the digit contained in a given image.

1.4.1 Computing training set accuracy

```
predict = predictOneVsAll(all_theta, TrainX);
fprintf('\nTraining Set Accuracy: %f\n', mean(double(predict == Trainy)) * 100);
```

```
Training Set Accuracy: 95.275000
```

1.4.2 Computing test set accuracy

```
predict = predictOneVsAll(all_theta, TestX);
fprintf('\nTest Set Accuracy: %f\n', mean(double(predict == Testy)) * 100);
```

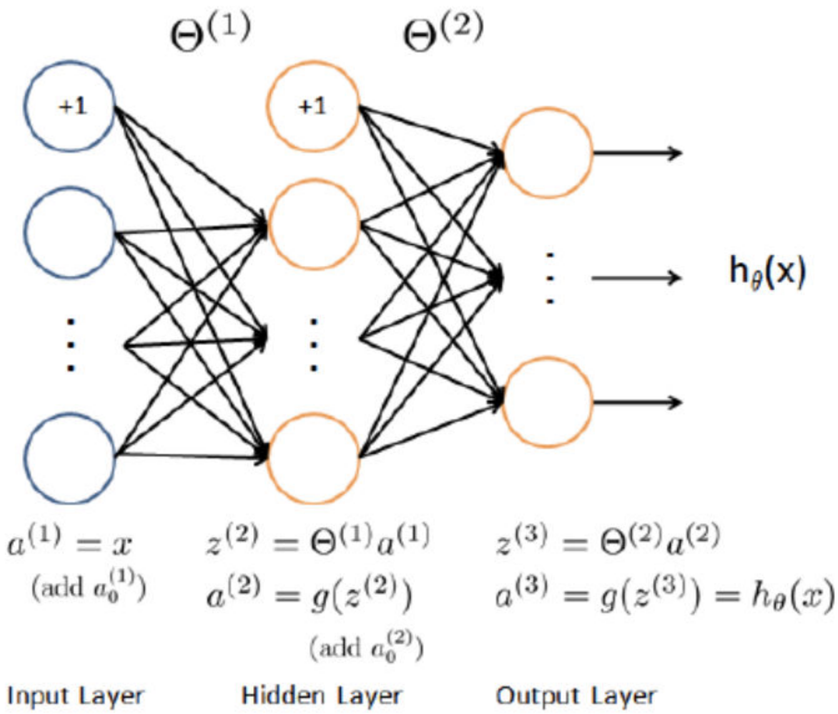
```
Test Set Accuracy: 91.500000
```

2. Neural Network

We will implement a neural network to recognize handwritten digits using the same training set as before. The neural network will be able to represent complex models that form non-linear hypotheses.

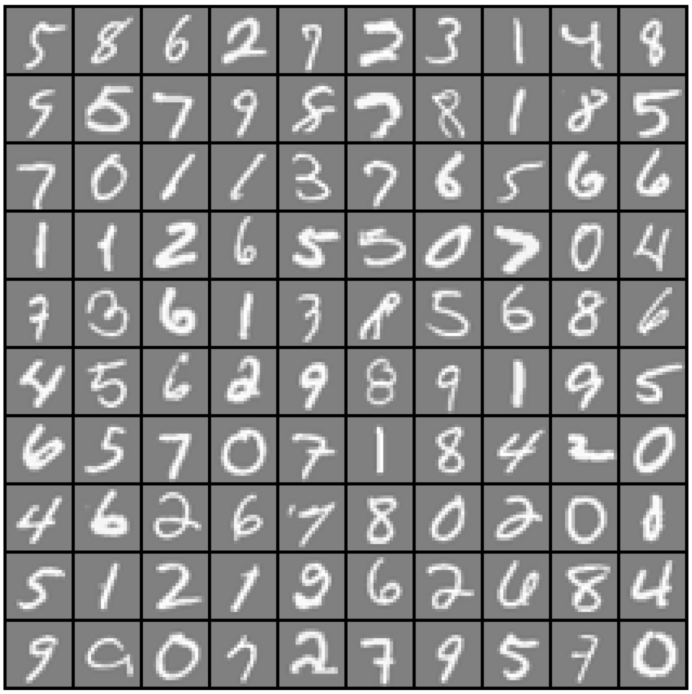
2.1 Model representation

Our neural network as it's shown below, It has 3 layers- an input layer, a hidden layer and an output layer. The inputs are pixel values of digit images. Since the images are of size 20 x 20, this gives us 400 input layer units.




```
clear

load('MNIST.mat'); % Load MNIST file
data = [X y];
data = data(randperm(size(data, 1)), :); % Shuffle the data set
% Split the data set : 80% training set, 20% test set
TrainX = data(1:4000,1:400);
Trainy = data(1:4000,401);
TestX = data(4001:end,1:400);
Testy = data(4001:end,401);
```

```
Trainm = size(TrainX,1); % Number of examples in training set
Testm = size(TestX,1); % Number of examples in test set
input_layer_size  = 400;  % 20x20 Input Images of Digits
hidden_layer_size = 25;   % 25 hidden units
num_labels = 10;          % 10 labels, from 1 to 10 (0 is mapped to 10)

% Randomly select 100 data points to display
rand_indices = randperm(size(TrainX, 1)); % Random permutation
select = rand_indices(1:100);
displayData(TrainX(select, :));
```



2.2 Regularized cost function

The cost function for neural networks with regularization is :

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left[-y_k^{(i)}\log((h_\theta(x^{(i)})_k) - (1 - y_k^{(i)})\log(1 - (h_\theta(x^{(i)}))_k)\right] + \frac{\lambda}{2m}\left[\sum_{j=1}^{25}\sum_{k=1}^{400}\left(\Theta_{j,k}^{(1)}\right)^2 + \sum_{j=1}^{10}\sum_{k=1}^{25}\left(\Theta_{j,k}^{(2)}\right)^2\right]$$

2.3 Backpropagation

2.3.1 Sigmoid derivate

The derivate for the sigmoid function can be computed as

$$g'(z) = \frac{d}{dz}g(z) = g(z)(1 - g(z))$$

where

$$\text{sigmoid}(z) = g(z) = \frac{1}{1 + e^{-z}}$$

2.3.2 Random initialization

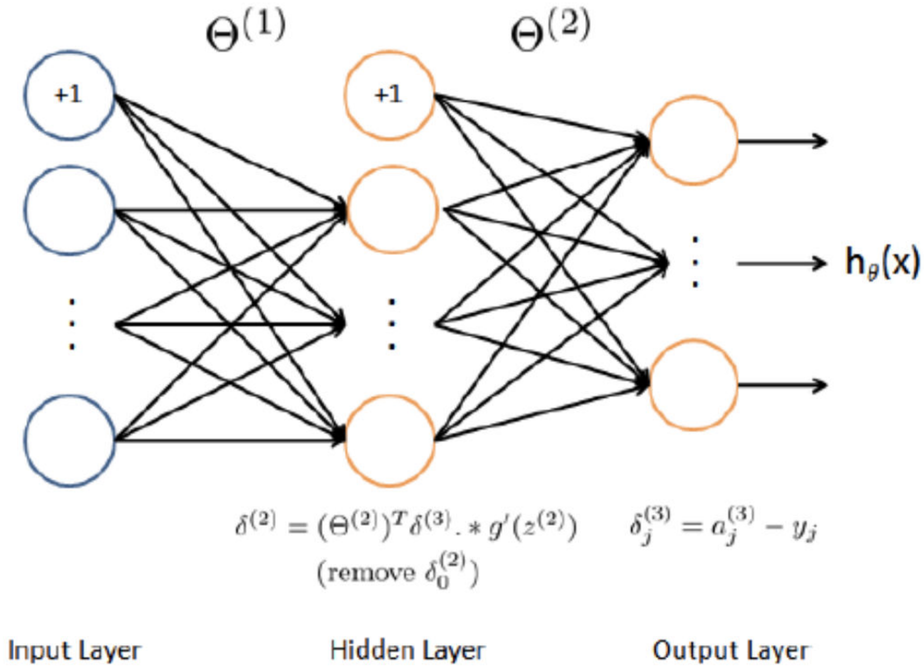
When training neural networks, it is important to randomly initialize the parameters for symmetry breaking. One effective strategy for random initialization is to randomly select values for $\Theta^{(l)}$ uniformly in the range $[-\epsilon_{init}, \epsilon_{init}]$. We use $\epsilon_{init} = 0.12$.

```
initial_Theta1 = randInitializeWeights(input_layer_size, hidden_layer_size);
initial_Theta2 = randInitializeWeights(hidden_layer_size, num_labels);

% Unroll parameters
initial_nn_params = [initial_Theta1(:) ; initial_Theta2(:)];
```

2.3.3 Backpropagation

The intuition behind the backpropagation algorithm is as follows. Given a training example $(x^{(t)}, y^{(t)})$, we will first run a 'forward pass' to compute all the activations throughout the network, including the output value of the hypothesis $h_\Theta(x)$. Then, for each node j in layer l, we would like to compute an 'error term' $\delta_j^{(l)}$ that measures how much that node was 'responsible' for any errors in our output.

For an output node, we can directly measure the difference between the network's activation and the true target value, and use that to define $\delta_j^{(3)}$ (since layer 3 is the output layer). For the hidden units, we compute $\delta_j^{(l)}$ based on a weighted average of the error terms of the nodes in layer $(l+1)$.



In detail, here is the backpropagation algorithm.

1. Set the input layer's values $(a^{(1)})$ to the t-th training example $x^{(t)}$. Perform a feedforward pass, computing the activations $(z^{(2)}, a^{(2)}, z^{(3)}, a^{(3)})$ for layers 2 and 3.
2. For each output unit k in layer 3 (the output layer), set $\delta_k^{(3)} = (a_k^{(3)} - y_k)$ where $y_k \in \{0, 1\}$ indicates whether the current training example belongs to class k $(y_k = 1)$, or if it belongs to a different class $(y_k = 0)$.
3. For the hidden layer $l = 2$, set $\delta^{(2)} = (\Theta^{(2)})^T\delta^{(3)}.*g'(z^{(2)})$
4. Accumulate the gradient from this example using the following formula: $\Delta^{(l)} = \Delta^{(l)} + \delta^{(l+1)}(a^{(l)})^T$.
5. Obtain the (unregularized) gradient for the neural network cost function by dividing the accumulated gradients by $\frac{1}{m}$: $\frac{\partial}{\partial\Theta_{ij}^{(l)}} J(\Theta) = D_{ij}^{(l)} = \frac{1}{m}\Delta_{ij}^{(l)}$
6. Add regularization to the gradient :

$$\frac{\partial}{\partial\Theta_{ij}^{(l)}} J(\Theta) = D_{ij}^{(l)} = \frac{1}{m}\Delta_{ij}^{(l)} \text{for}\, j = 0,$$
$$\frac{\partial}{\partial\Theta_{ij}^{(l)}} J(\Theta) = D_{ij}^{(l)} = \frac{1}{m}\Delta_{ij}^{(l)} + \frac{\lambda}{m}\Theta_{ij}^{(l)} \text{for}\, j \geq 1$$

2.3.4 Learning parameters using `fmincg`

We use a MATLAB built-in function called `fmincg` instead of taking gradient descent steps to find the optimal parameters.

```
options = optimset('MaxIter', 50);
lambda = 1;
```

```
% Create short-hand for the cost function
costFunction = @(p) neuralNetworkCostFunction(p, input_layer_size, hidden_layer_size, n
[nn_params, ~] = fmincg(costFunction, initial_nn_params, options);
```

```
Iteration     1 | Cost: 3.302155e+00
Iteration     2 | Cost: 3.258978e+00
Iteration     3 | Cost: 3.215418e+00
Iteration     4 | Cost: 3.110598e+00
Iteration     5 | Cost: 2.960005e+00
Iteration     6 | Cost: 2.580039e+00
Iteration     7 | Cost: 2.389873e+00
Iteration     8 | Cost: 2.156498e+00
Iteration     9 | Cost: 1.976863e+00
Iteration    10 | Cost: 1.755101e+00
Iteration    11 | Cost: 1.571259e+00
Iteration    12 | Cost: 1.397567e+00
Iteration    13 | Cost: 1.280110e+00
Iteration    14 | Cost: 1.225921e+00
Iteration    15 | Cost: 1.130711e+00
Iteration    16 | Cost: 1.087208e+00
Iteration    17 | Cost: 1.054423e+00
Iteration    18 | Cost: 1.002721e+00
Iteration    19 | Cost: 9.406952e-01
Iteration    20 | Cost: 9.157846e-01
Iteration    21 | Cost: 8.947648e-01
Iteration    22 | Cost: 8.671386e-01
Iteration    23 | Cost: 8.072407e-01
Iteration    24 | Cost: 7.680633e-01
Iteration    25 | Cost: 7.235632e-01
Iteration    26 | Cost: 6.986662e-01
Iteration    27 | Cost: 6.912245e-01
Iteration    28 | Cost: 6.703156e-01
Iteration    29 | Cost: 6.609040e-01
Iteration    30 | Cost: 6.535368e-01
Iteration    31 | Cost: 6.408100e-01
Iteration    32 | Cost: 6.279499e-01
Iteration    33 | Cost: 6.219965e-01
Iteration    34 | Cost: 6.172341e-01
Iteration    35 | Cost: 6.092191e-01
Iteration    36 | Cost: 6.015814e-01
Iteration    37 | Cost: 5.938516e-01
Iteration    38 | Cost: 5.880576e-01
Iteration    39 | Cost: 5.821867e-01
Iteration    40 | Cost: 5.748268e-01
Iteration    41 | Cost: 5.616795e-01
Iteration    42 | Cost: 5.532814e-01
Iteration    43 | Cost: 5.450111e-01
Iteration    44 | Cost: 5.369558e-01
Iteration    45 | Cost: 5.274277e-01
Iteration    46 | Cost: 5.220770e-01
Iteration    47 | Cost: 5.154129e-01
Iteration    48 | Cost: 5.120034e-01
Iteration    49 | Cost: 5.083241e-01
Iteration    50 | Cost: 5.045486e-01
```

```
% Obtain Theta1 and Theta2 back from nn_params
Theta1 = reshape(nn_params(1:hidden_layer_size * (input_layer_size + 1)), hidden_layer_
Theta2 = reshape(nn_params((1 + (hidden_layer_size * (input_layer_size + 1))):end), nu
```

2.4 Computing accuracy

```
% Training set accuracy
```

```
pred = predictNeuralNetwork(Theta1, Theta2, TrainX);
fprintf('\nTraining Set Accuracy: %f\n', mean(double(pred == Trainy)) * 100);
```

```
Training Set Accuracy: 95.450000
```

```
% Test set accuracy
pred = predictNeuralNetwork(Theta1, Theta2, TestX);
fprintf('\nTest Set Accuracy: %f\n', mean(double(pred == Testy)) * 100);
```

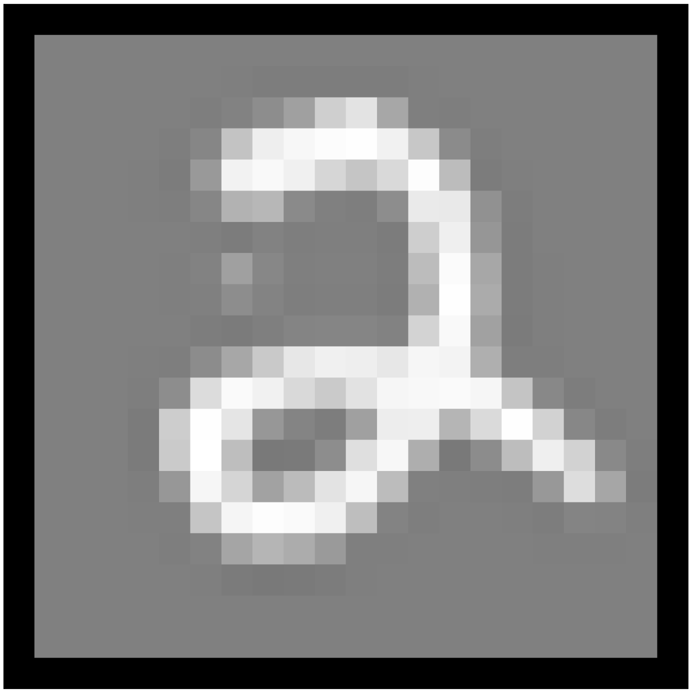
```
Test Set Accuracy: 92.600000
```

2.5 Prediction test

```
random_example = randi(Testm); % Draw a random example in test set
% Predict
pred = predictNeuralNetwork(Theta1, Theta2, TestX(random_example,:));
fprintf('\nNeural Network Prediction: %d \nDigit Label: %d\n', mod(pred, 10), mod(Testy
```

```
Neural Network Prediction: 2
Digit Label: 2
```

```
% Display
displayData(TestX(random_example, :));
```

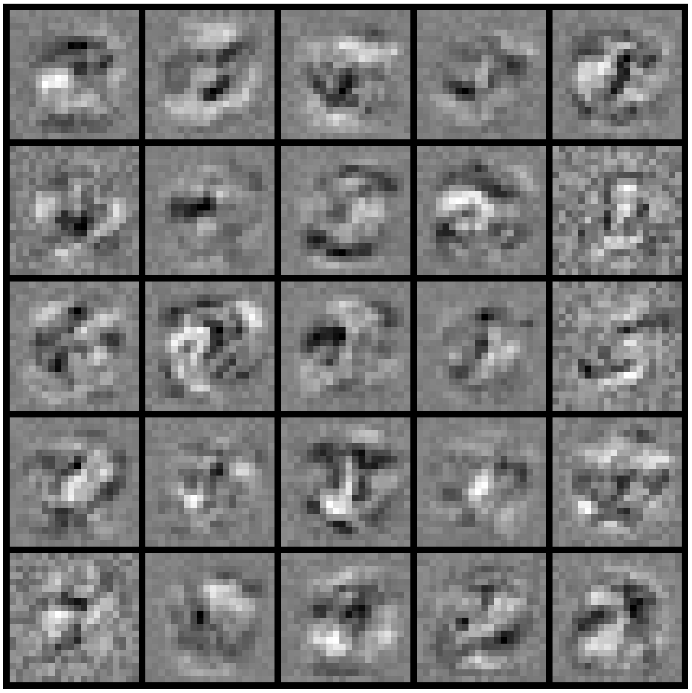


2.6 Visualizing the hidden layer

It shows an image with 25 units, each corresponding to one hidden unit in the network.

```
% Visualize Weights
```

```
displayData(Theta1(:, 2:end));
```



We find that the hidden units corresponds roughly to detectors that look for strokes and other patterns in the input.