Project de MOOC

Introductions:

As there is not project in this course, I selected one task in another Machine Learning course. In this project, we are supposed to implement hand-written digit recognition by constructing a neural network. There are

5000 training examples in data file, where each training example is a

20 pixels by 20 pixels 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. Each

of these training examples becomes a single row in our data matrix. This

gives us a 5000 by 400 matrix where every row is a training example for a

handwritten digit image. The second part of the training set is a 5000-dimensional vector y which contains labels for the training set.

Solution:

In this project, we used a neural network with 3 layers to recognize hand-written digits. The input layer is 400 dimensions, the hidden layer is 25 dimensions and the output layer is 10 dimensions. Before c a constructing a neural network, we need to implement a few algorithms.

1.Implement the feedforward algorithm to get the cost function:

Firstly, we follow the instructions below to compute h\_theta(x)

z2 = theta1 \* a1;

a2 = g(z2);

z3 = theta2 \* a2;

h\_theta(x) = a3 = g(z3);

aij : activation of unit i in layer j

thetaj : matrix of of weights controlling function mapping from layer j to layer j+1.

g is the sigmoid function: g(x) = 1/(1 + exp(-x))

And then we compute the cost function. The cost is:

regularized cost function (need to replace)

2. Implement the back propagation algorithm to get the gradient.

The instruction is as follows:

Training set {(x(1), y(1)), (x(2), y(2)), … , (x(m), y(m))}

Set XX = 0 (for all l, I, j)

for i = 1 to m:

Set xx = xx;

Perform forward propagation to compute xx for l = 2, 3, …, L

Using xx, computer xxxxxx

Compute xxxxxxxxxxxxxxxxxxxxxxxxxxx

Xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Xxxxxx

And now we’re able to constructing the neural network. Steps are as follows:

1. Randomly initialize weights.
2. Implement forward propagation to get xxx for any xxx.
3. Compute cost function J(xx).
4. Implement back propagation to compute partial derivatives xxxx.
5. Use gradient checking to compare xxx computed using back propagation with the one using numerical estimate of gradient of J(xxx). And then, disable gradient checking (this is very important).
6. Use gradient descent to try to minimize J(xxx) as a function of parameter xxx.