CS 273A Machine Learning Homework 2

Chen Li

October 18, 2015

1 Problem1

1.1 Exactly One On

1. The dimension of each layer is 5, 7, 1. In my practice, the number of hidden(when greater than the input layer) does not make a big difference.
2. I use the mini-batch stochastic gradient descent method, where I choose B to be 100.
3. I set the weight decay term $\lambda = 0.001$
4. The error function I use is $E(w) = \frac{1}{2}\sum\limits_{n=1}^{N}\{y(x_n, w) - t_n\}^2$
5. When the output of neural network is less than 0.5, I classify it as 0, and vice versa.
6. The learning rate in this problem is tricky. When I set a fixed learning rate, the result is very unstable (probably the global optimum of the error function is hard to find). Then I choose the variable learning rate, i.e., I set a fix learning rate $\alpha = 0.1$, and when update the W, I use $\alpha^* = \frac{\alpha}{gradient_{mini-batch}}$. This approach magically works, I can result 100% in less than 3000 iterations.

1.2 Autocoder

1. The dimension I choose is 100, 30 and 100.
2. Same mini-batch SGD choice as above
3. The error function I use is $E(w) = \frac{1}{N}\sum\limits_{n=1}^{N}\|y(x_n, w) - t_n\|$
4. This time, because the curve seems easy to optimize, I fix the alpha.

2 Problem2

1. I use the same image generation strategy as in HW1, except this time I fix the number of on pixels, and noise levels (the α in HW1). Also, the picture size now become 5×7.
2. I use the exactly same setting as Exactly one on problem, except that this time the dimension of each layer is 35,50,1

3 Implementation

In *hw2.py*, each problem has its own data generation function, train function and test function. The $Test()$ method performs the test on different sets of parameters. The results locate in the $test_results$ folder. The Hinton of the best network of the three problems locate in *diagrams* folder. Note that my Hinton diagram choose the best performance parameters sets, and it use black and white indicate positive and negative, and use the size of the rectangle to indicate the absolute value of the weight. Also, I concatenate two weight matrix of a neural network into one. with the shape being $(hidden_dim + 1) \times (input_dim + 1 + output_dim)$, i.e, transpose the second layer weight matrix and concatenate it on the right of the first layer weight matrix.