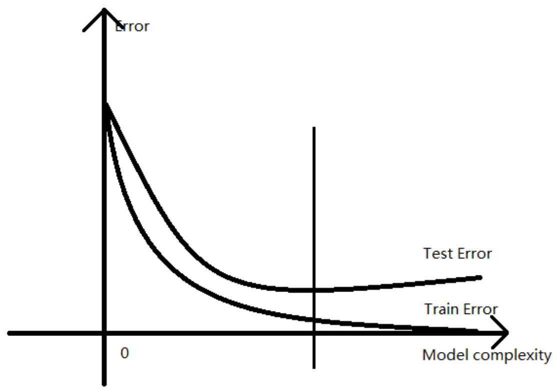
10-707: Deep Learning, Fall 2017

Homework 1

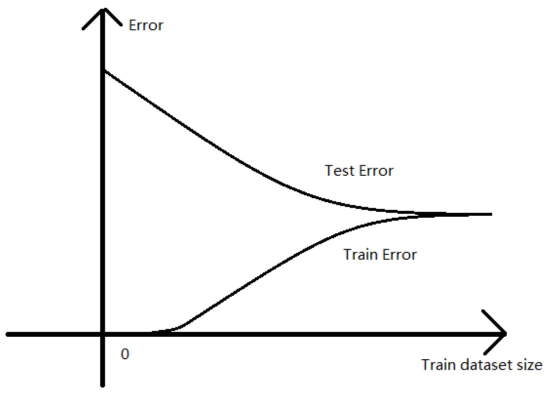
Lecturer: Russ Salakhutdinov *Name: Yuan Liu(yuanl4)*

1 Problem 1

1.1 Error vs model complexity



1.2 Error vs size of training dataset



1.3 Early stopping

Early stopping is a reasonable regularization metric. Machine learning algorithms train a model based on a finite set of training data with an iterative method. The goal of machine learning is to predict the unseen observations. Up to a point, the iterative update methods improves the model's performance on unseen observations. Past that point, overfitting occurs. Early stopping rules provide guidance as to how many iterations can be run before the learner begins to over-fit.

2 Problem 2

- K-nearest-neighbor regression:

  For knn, the estimator is given by:

  $$\hat{f}(x^*) = \frac{1}{k} \sum_{i \in N_k(x^*)} y_i$$

  $N_k(x^*)$ contains the indices of the k closest points of $x_1, \ldots, x_N$ to x^*. Then we can know:

  $$l_i(x^*; \mathcal{X}) = \begin{cases} \frac{1}{k}, & i \in N_k(x^*) \\ 0, & otherwise \end{cases} \tag{1}$$

- Linear regression: For linear regression, the estimator is given by:

  $$\hat{f}(x^*) = {x^*}^T w$$

  where $w = (X^T X)^{-1} X^T y$, $y = (y_1, \ldots, y_N)^T$ and $X = (x_1, \ldots, x_N)^T$. Then we can get:

  $$\hat{f}(x^*) = {x^*}^T (X^T X)^{-1} X^T y$$

  So

  $$l_i(x^*; \mathcal{X}) = ({x^*}^T (X^T X)^{-1} X^T)_i$$

  $l_i(x^*; \mathcal{X})$ equals to the i_{th} component of ${x^*}^T (X^T X)^{-1} X^T$.

3 Problem 3

- Normalization: $p(x = 1|\mu) + p(x = -1|\mu) = \frac{1+\mu}{2} + \frac{1-\mu}{2} = 1$
- Mean: $E[x] = 1 * p(x = 1|\mu) + (-1) * p(x = -1|\mu) = \mu$
- Variance: $Var[x] = E[x^2] - (E[x])^2 = \frac{1+\mu}{2} + \frac{1-\mu}{2} - \mu^2 = 1 - \mu^2$
- Entropy: $Entropy = -\sum_{i \in \{-1,1\}} p(x = i|\mu) \log p(x = i|\mu) = -\frac{1+\mu}{2} \log \frac{1+\mu}{2} - \frac{1-\mu}{2} \log \frac{1-\mu}{2}$

4 Problem 4

Denote l is the correct label of x, and t is the label of x given by the dataset. So we can have the following formula:

$$p(l=1|t=1)=1-\epsilon$$

$$p(l=1|t=0)=\epsilon$$

$$\begin{aligned} p(l=1|x;w) &= p(t=1|x;w)p(l=1|t=1)+p(t=0|x;w)p(l=1|t=0) \\ &= y(x,w)(1-\epsilon)+(1-y(x,w))\epsilon \end{aligned}$$

Then we can get:

$$l \sim \mathit{Bernoulli}\left(y(x,w)(1-\epsilon)+(1-y(x,w))\epsilon\right)$$

$$p(l|x,w)=\left[y(x,w)(1-\epsilon)+(1-y(x,w))\epsilon\right]^{l}\left[1-y(x,w)(1-\epsilon)-(1-y(x,w))\epsilon\right]^{1-l}$$

$$\begin{aligned} \text{cost function} &= -\log p(l|x,w) \\ &= -l\log(y(x,w)(1-\epsilon)+(1-y(x,w))\epsilon)-(1-l)log(1-y(x,w)(1-\epsilon)-(1-y(x,w))\epsilon) \\ &= -l*\log(y-2y\epsilon+\epsilon)-(1-l)*\log(1-y+2y\epsilon-\epsilon) \end{aligned}$$

Where l is the label of training dataset, y is the output of neural network.

If $\epsilon=0$, then

$$\text{cost function} = -l*\log y-(1-l)*log(1-y)$$

which is the standard negative log likelihood of binary classification.

5 Problem 5

First represent two networks in the following form:

- Sigmoid network: Input $x=(x_1,\ldots,x_p)^T$, First Layer: $a^{sig}=W_1^{sig}x+b_1^{sig}$, Activation function: $h^{sig}=\sigma(a^{sig})$, Second Layer: $o^{sig}=W_2^{sig}h^{sig}+b_2^{sig}$
- Tanh network: Input $x=(x_1,\ldots,x_p)^T$, First Layer: $a^{tanh}=W_1^{tanh}x+b_1^{tanh}$, Activation function: $h^{tanh}=tanh(a^{tanh})$, Second Layer: $o^{tanh}=W_2^{tanh}h^{tanh}+b_2^{tanh}$

By observation we can have:

$$\sigma(2a)=\frac{tanh(a)+1}{2}$$

First, we can assume:

$$W_1^{sig}=2W_1^{tanh}, \quad b_1^{sig}=2b_1^{tanh}$$

Then

$$a^{sig}=W_1^{sig}x+b^{sig}=2W_1^{tanh}x+2b_1^{tanh}=2a^{tanh}$$

$$h^{sig}=\sigma(2a^{tanh})=\frac{tanh(a^{tanh})+1}{2}=\frac{h^{tanh}+1}{2}$$

Second, we can assume:

$$W_2^{sig} = 2W_2^{tanh}, \quad b_2^{sig} = b_2^{tanh} - W_2^{tanh} \cdot \mathbf{1}$$

Where **1** is a vector and all its component is 1.Then

$$o^{sig} = W_2^{sig} h^{sig} + b_2^{sig} = 2W_2^{tanh} \frac{h^{tanh} + \mathbf{1}}{2} + b_2^{tanh} - W_2^{tanh} \cdot \mathbf{1} = o^{tanh}$$

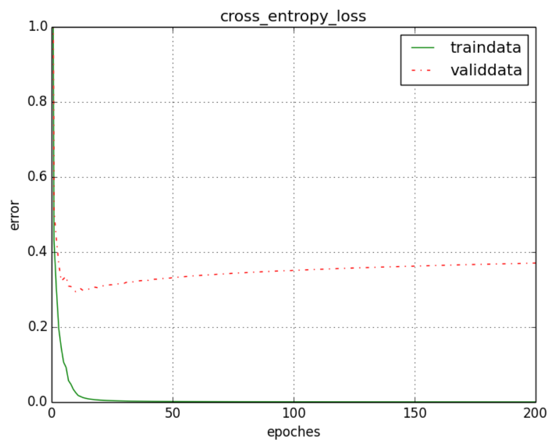
As a result, we can have the following equation:

$$W_1^{sig} = 2W_1^{tanh}, \quad b_1^{sig} = 2b_1^{tanh}$$

$$W_2^{sig} = 2W_2^{tanh}, \quad b_2^{sig} = b_2^{tanh} - W_2^{tanh} \cdot \mathbf{1}$$

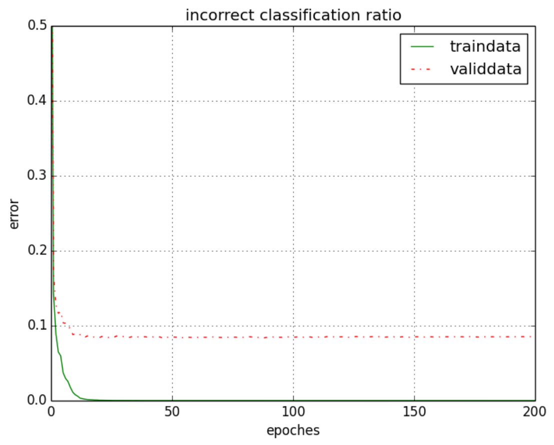
6 Problem 6

6.1 (a)



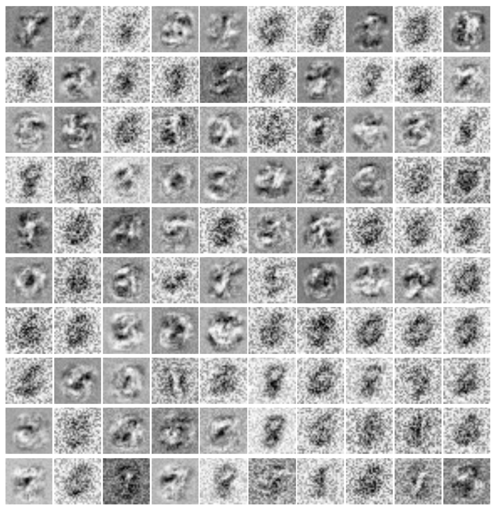
When the training is going on, the training error will keep decreasing. The validation error will first decrease, then up to a point, begins to increase.
The network's performance keeps becoming better and better on the training dataset. The network's performance on the validation dataset will first become better then becomes worse, because of over-fitting.

6.2 (b)



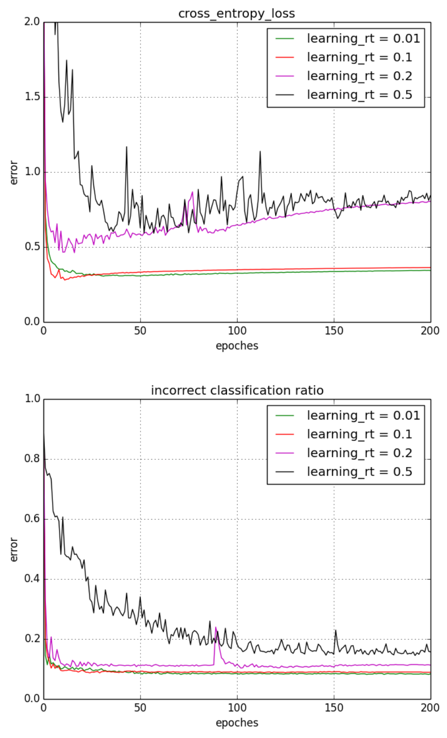
The over-fitting is difficult to be observed by the classification error.

6.3 (c)

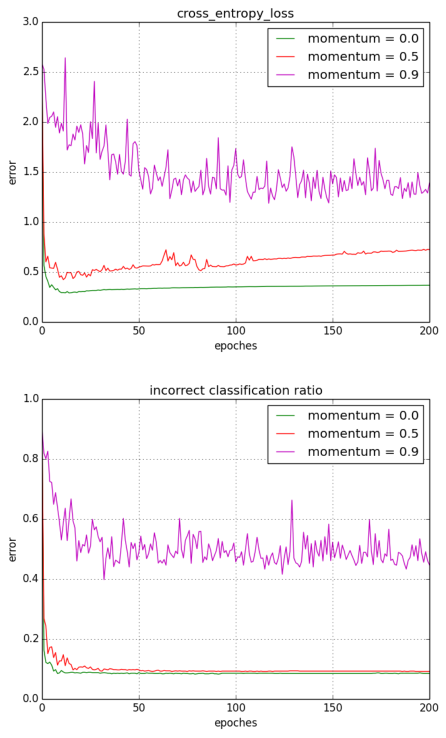


For example the sub-image on row 10 column 9 looks like number 1, the sub-image on row 4 column 5 looks like number 3, the sub-image on row 9 column 7 looks like number 8. I think this model can learn some structured features.

6.4 (d)



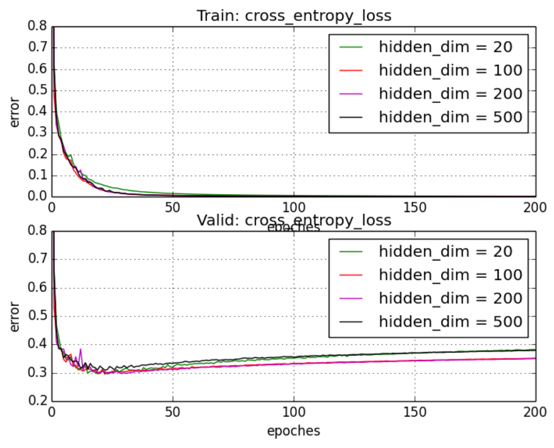
(a) Problem d: different learning rate



(b) Problem d: different momentum

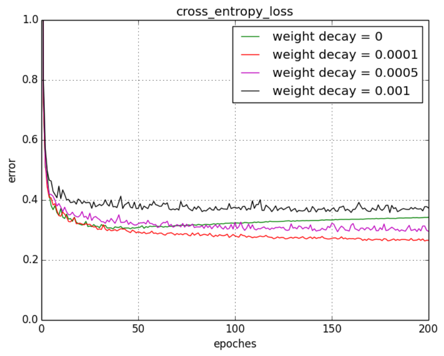
For this case, decreasing the learning rate will improves the model's performance and the momentum makes the model's performance worse. So I will choose learning rate equals 0.01 and momentum equals 0

6.5 (e)



$Hidden\ dimension = 100\ or\ 200$ is most suitable for this task. When the number of hidden units is too little or too much, the performance of network will become worse.

6.6 (f)



(c) Problem f: different L2 coefficients

(d) Problem f: visualization

According to problem (d) and (e), we found

$$learning\ rate = 0.01$$

$$momentum = 0$$

$$hidden\ dimension = 100$$

is the most suitable parameter for this task. Then I tried some L2 regularization coefficients and find out

$$L2\ coefficient = 0.0001$$

$$epoch\ number = 180$$

is the best choice.

$$train\_loss\_cross\_entropy = 0.03, valid\_loss\_cross\_entropy = 0.27, test\_loss\_cross\_entropy = 0.33$$

$$train\_loss\_\%incorrect = 0.0, valid\_loss\_\%incorrect = 0.08, test\_loss\_\%incorrect = 0.09$$

6.7 (g)

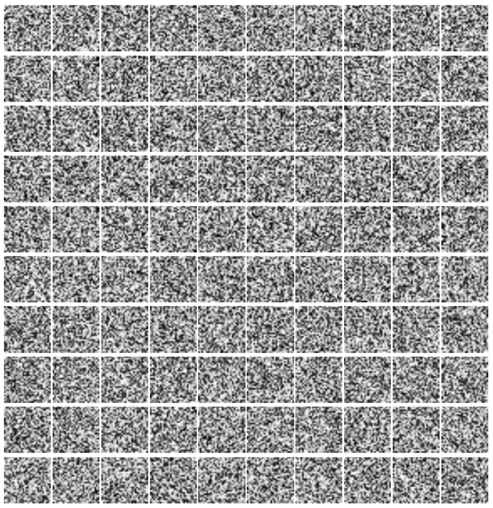
After cross validation, we find out the following parameters.

$$learning\ rate = 0.01$$

$$momentum = 0$$

$$L2\ coefficient = 0$$

$$epoch\ number = 140$$



I cannot find out any structure in the visualization of first layer. It's much worse than the ones that I obtained when training a single-layer network.

The 1-layer network outperform a 2-layer model in term of generalization capabilities.

$$train\_loss\_cross\_entropy = 0.32, valid\_loss\_cross\_entropy = 0.50, test\_loss\_cross\_entropy = 0.55$$

$$train\_loss\_\%incorrect = 0.10, valid\_loss\_\%incorrect = 0.15, test\_loss\_\%incorrect = 0.17$$

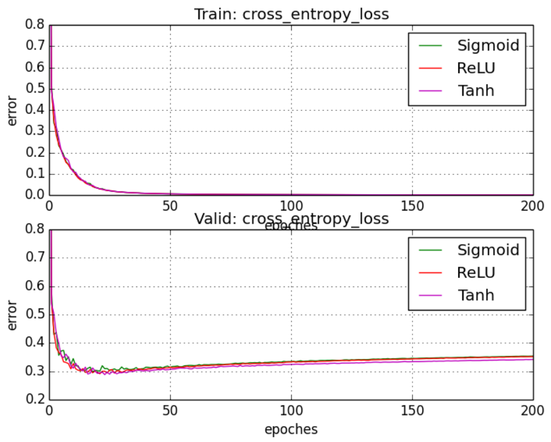
6.8 (h)

Batch normalization helps in terms of speed and accuracy. Before I use batch normalization, I need to set $learning\_rate = 0.01$ and train 140 epochs. After I use batch normalization, I can set $learning\_rate = 0.1$ and only train 7 epochs and get better result.

$$train\_loss\_cross\_entropy = 0.12, valid\_loss\_cross\_entropy = 0.36, test\_loss\_cross\_entropy = 0.53$$

$$train\_loss\_\%incorrect = 0.04, valid\_loss\_\%incorrect = 0.12, test\_loss\_\%incorrect = 0.14$$

6.9 (i)



I use three different activation function to train the model. The performances of Sigmoid and ReLu are almost the same, and Tanh performs slightly better than them.