DD2424 - Assignment 2

Thi Thuy Nga Nguyen

1 Gradient Computation

1.1 Gradient check

The numerically computed gradients, g_n, and the analytically computed gradients, g_a, were compared by using relative error

$$error = \frac{|g_a - g_n|}{max(\epsilon, |g_a| + |g_n|)},$$

where $\epsilon = 10^{-30}$. The initial parameters, the bias vectors were set 0 and the weight matrices W were randomly drawn from a Gaussian distribution with mean 0 and standard deviation $\frac{1}{\sqrt{d}}$ for layer 1 and $\frac{1}{\sqrt{m}}$ for layer 2, where d is dimension of input and $m = 50$ is the number of the nodes in the hidden layer.
In the result, for first 5 images in data_batch_1 in a reduced dimension (20) and $h = 1e - 5$, we get:

```
The number of errors (relative error > 1e-6) of W: 0
The maximum of relative error of W: 2.673371486592694e-07
The number of errors (relative error > 1e-6) of b: 0
The maximum of relative error of b: 2.459301143699773e-08
```

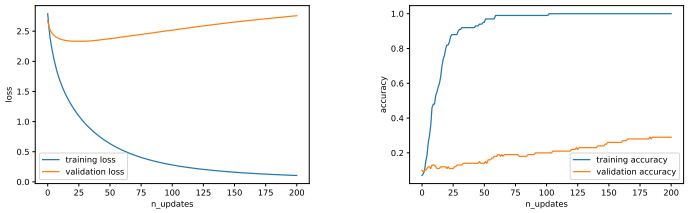
The relative errors being almost very small show the precision for the numerical gradients.

1.2 Sanity check

I trained my network on a small set of training data which contains first 100 examples in data_batch_1 for training and first 100 examples in data_batch_2 for validation with $\lambda = 0$, $\eta = 0.01$, the batch size is 100 and 200 epochs. The results in figure 1 show a very low loss on training set and overfiting in training set where the accuracy in validation set is too low comparison to the one in training set.

2 Cyclical learning rate

In this part, I trained the network with mini-batch gradient descent using the data in the file data_batch_1 for training, the file data_batch_2 for validation and the file



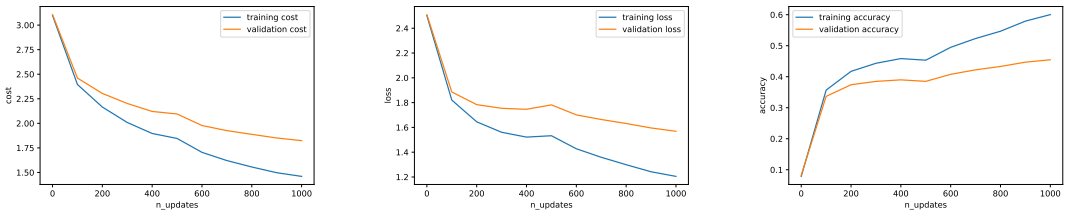
(a) (b)

Figure 1: The loss (a), accuracy (b) after each updating with $\lambda = 0, n\_epochs = 200, n\_batch = 100, \eta = 0.01$.

test_batch for testing. The data was centralized and normalized by mean and standard deviation of training data.
For the initial parameters, the bias vectors b were set 0 and the weight matrices W were randomly drawn from a Gaussian distribution with mean 0 and standard deviation $\frac{1}{\sqrt{d}}$ for layer 1 and $\frac{1}{\sqrt{m}}$ for layer 2, where $d = 3072$ is dimension of input and $m = 50$ is the number of the nodes in the hidden layer.

2.1 Train network with cyclical learning rate

The cyclical learning rate was triangular with eta_min = 1e-5, eta_max = 1e-1 and step size $n\_s = 500$. With the batch size of 100, $\lambda = 0.01$ and one full cycle (corresponding to 10 epochs) I got the test accuracy of 46.42% and the loss, cost, accuracy curves of the training and validation set after every 100 updates as in figure 2,



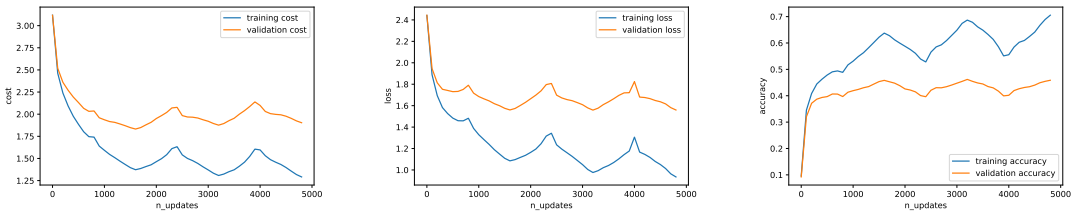
(a) (b) (c)

Figure 2: The cost (a), loss (b) and accuracy (c) after each updating after one cycle of training with $\lambda = 0.01, n\_batch = 100, eta\_min = 1e-5, eta\_max = 1e-1, n\_s = 500$ and $n\_epochs = 10$.

The figure 2 shows when the learning rate increases and reaches to its maximum value, the cost decreases slower, the loss seems increasing and the accuracy increases slower and seems decreasing. But after that, when η starts decreasing from this value, the loss and the cost curve decrease and the accuracy increases more rapidly.

2.2 Train network for real

I next ran the training for 3 cycles with a larger step size $n\_s = 800$. In the result, I got the test accuracy of 47.16% and the loss, cost, accuracy curves of the training and validation set after every 100 updates as in figure 3,



(a) (b) (c)

Figure 3: The cost (a), loss (b) and accuracy (c) after each updating after 3 cycles of training with $\lambda = 0.01, n\_batch = 100, eta\_min = 1e-5, eta\_max = 1e-1, n\_s = 800$.

The figure 3 shows when the learning rate increases and reaches to its maximum value in each cycle the cost, the loss increase to some picks and the accuracy decreases to some bottoms. But after that, when η starts decreasing to its minimum value, the loss and the cost curve decrease to lower values comparing to the previous lowest points and the accuracy increases to a higher value comparison with the last highest. When the learning rate reaches its minimum value, the loss/cost goes down to a local minimum and the accuracy goes to a pick in both training and validation set.

3 Random search

I next constructed a coarse-to-fine random search for λ. In this part, I used all the data for training (5 batches) which contains 50000 images except for 5000 ones for validation. The value of λ was uniformly taken in the range of $10^{l_{\min}}$ to $10^{l_{\max}}$.

3.1 Coarse search

I trained the network for 2 cycles, n_s = 2 floor(n / n_batch) with 20 different values of λ being uniformly sampled in the range of $10^{l_{\min}}$ to $10^{l_{\max}}$ with $l_{\min} = -5$, $l_{\max} = -1$. I got 3 best performance networks:

- the validation accuracy is 52.86% with λ = 7.5578e-4,
- the validation accuracy is 52.76% with λ = 4.165542e-3,
- the validation accuracy is 52.62% with λ = 1.07751e-3,

and other hyper-parameter n_batch = 100, n_epochs = 8, eta_min = 1e-5, eta_max = 0.1, n_s = 900, n_cycle = 2, n_nodes = 50 for hidden layer.

3.2 Fine search

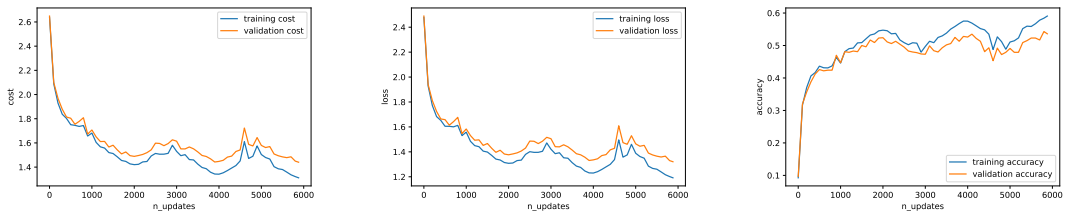
Narrowing the range to $l_{\min} = -4$, $l_{\max} = -2$, I trained the network for 3 cycles with 20 different values of λ being uniformly sampled in the range of $10^{l_{\min}}$ to $10^{l_{\max}}$ and got 3 best performance networks:

- the validation accuracy is 52.8% with λ = 2.68647e-3,
- the validation accuracy is 52.56% with λ = 4.8179e-4,
- the validation accuracy is 52.54% with λ = 6.51218e-3,

and other hyper-parameter n_batch = 100, n_epochs = 12, eta_min = 1e-5, eta_max = 0.1, n_s = 900, n_cycle = 3, n_nodes = 50 for hidden layer.

3.3 Train network with the best performance hyper-parameter

I next trained the network for 3 cycles with the value of λ = 2.68647e-3 corresponding to the best performance obtained from fine search. All data was used except 1000 images for validation. In the result, I got the test accuracy is 52.01% and the loss, cost, accuracy curves for training and validation set after every 100 updates in figure 4.



(a) (b) (c)

Figure 4: The cost (a), loss (b) and accuracy (c) after each updating after 3 cycles of training with $\lambda = 2.68647e-3, n\_batch = 100, eta\_min = 1e-5, eta\_max = 1e-1, n\_s = 980$.