Deep Learning in Data Science, DD2424

Short report on lab assignment 2

Two-layer neural network with multiple outputs

Hamid Dimyati

1. Introduction

The goal of this lab assignment 2 is to train and test a two-layer neural network to classify multiple classes of CIFAR-10 datasets. The network applies cross-entropy loss function and L_2 regularization with parameters learning using mini-batch gradient descent.

The tool used for this assignment is primarily `python 3.7.3`, along with several packages including `numpy 1.16.4`, `pandas 0.25.3`, and `matplotlib 3.1.0`.

2. Results and discussion

Before start building the network, a standardization (Z-score normalization) is performed. To validate whether the self-build codes, particularly the function to compute the gradient of parameters $\boldsymbol{W} = \{W_1, W_2\}$, and $\boldsymbol{b} = \{b_1, b_2\}$, could run appropriately based on the mathematical functions, comparing the results with alternative ways using the numerical approach as provided in the instruction documents is necessary. A `numpy.allclose` function is used to ensure the difference between those two outputs is still tolerable, which follows the equation (1):

$$\frac{|g_a - g_n|}{max(eps, |g_a| + |g_n|)} \approx absolute\ tolerance \quad (1)$$

Using my codes (g_a), the gradient of $\boldsymbol{W}$ could produce similar results to the numerical approach (g_n) with $absolute\ tolerance = 1e-3$, while the gradient of $\boldsymbol{b}$ could successfully achieve similar results to the numerical approach (g_n) with $absolute\ tolerance = 1e-4$. Training the network on 100 samples of the training data with regularization turned off (lambda=0), η equal to $1e-2$, 200 epochs, and 50 hidden nodes, results in minimal cost value as depicted in Figure 1.

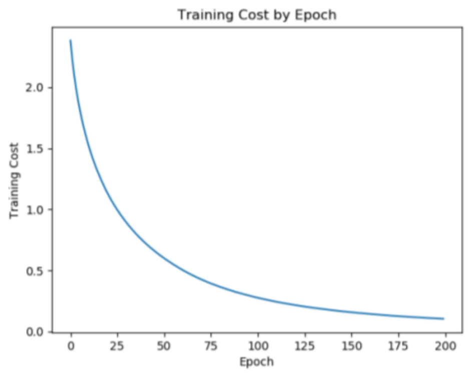


Figure 1. Cost value as a function of epoch

2.1. Applying cyclical learning rate

Once the codes were proven to yield reliable results, the two-layer neural network employing cyclical learning rate (CLR) is then trained based on different hyperparameters, as stated in Table 1.

Table 1. Hyperparameter settings for two models

Model	**Hyperparameters**						
	Total epoch	**Batch size**	**Lambda**	**Eta min**	**Eta max**	**Step size**	**Cycle**
Model 1	10	100	1e-2	1e-5	1e-1	500	1
Model 2	48	100	1e-2	1e-5	1e-1	800	3

The datasets mainly used for this experiment are CIFAR-10 batch 1 for training, CIFAR-10 batch 2 for validation, and CIFAR-10 test batch for testing. The detail results are seen in Table 2 and two figures below. Based on those figures, when $t = 2ln_s$ or $\eta_t = \eta_{min}$, the cost value could go down at its lowest point during the cycle. It is also understood that applying cyclical learning rates with unsuitable lambda value could end up with overfitting, as seen in Figure 2 and Figure 3.

Table 2. Accuracy for each model

Model	**Accuracy**			**Accuracy gap (train – test)**
	Training	**Validation**	**Testing**	
Model 1	60.92%	45.58%	46.06%	14.86%
Model 2	71.59%	47.04%	47.61%	23.98%

Model 1

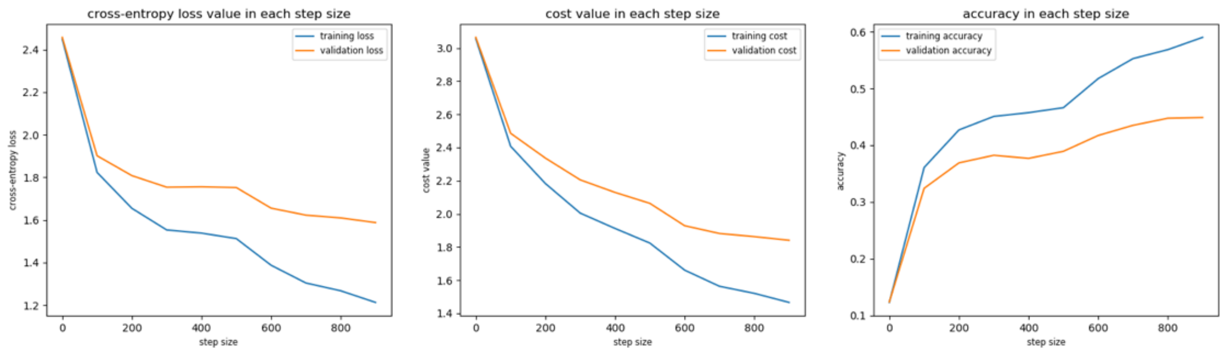


Figure 2. Cross-entropy loss (left), cost value (middle), and accuracy (right) as a function of the step size from Model 1

Model 2

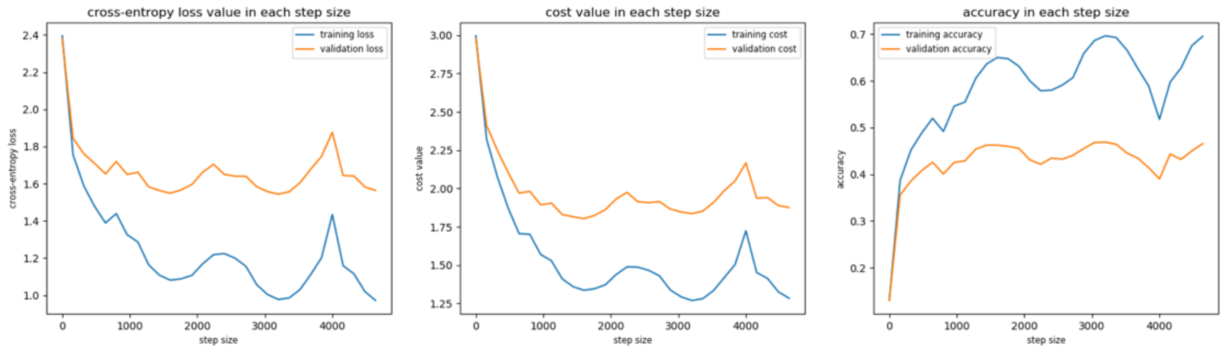


Figure 3. Cross-entropy loss (left), cost value (middle), and accuracy (right) as a function of the step size from Model 2

2.2. Coarse-to-fine random search

This experiment aims to find the optimal regularization parameter since the two models above seem to experience overfitting. This technique is beneficial to find the best hyperparameters through repetitively randomly selecting a value, which falls between the given optimal range. The datasets mainly used for this experiment are CIFAR-10 batch 1 to 5 except the last 5,000 samples for training, the last 5,000 samples in CIFAR-10 batch 5 for validation, and CIFAR-10 test batch for testing. In the coarse search, I define list of lambda value to be tested: $[10^{-5}, 10^{-4.5}, 10^{-4}, 10^{-3.5}, 10^{-3}, 10^{-2.5}, 10^{-2}, 10^{-1.5}, 10^{-1}]$, batch size 100, step size 900 and for 2 cycles learning. Below are the top three lambdas during the coarse search:

Table 3. Accuracy for the top three lambdas during coarse search

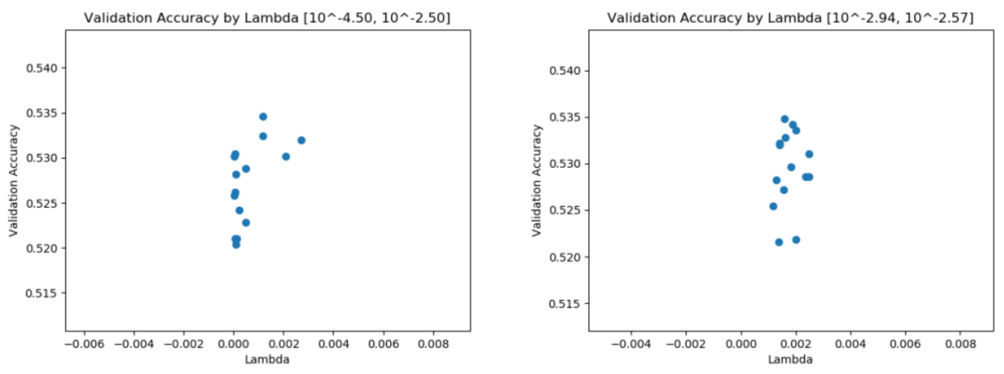
Lambda	Validation Accuracy
$10^{-4.5}$	52.56%
$10^{-2.5}$	52.50%
10^{-3}	52.36%

$$l = l_{min} + (l_{max} - l_{min}) * rand(1,1); \qquad (2)$$
$$lambda = 10^{l}$$

In fine search, I use 3 cycles learning and iteratively generate 15 random lambda values within the range of $[10^{-4.5}, 10^{-2.5}]$ using equation (2), selecting the top three lambda values based on its validation accuracy, update the range based on the top three lambdas until it converges, as the process could be seen in Table 4 and Figure 4 below. After three iterations, the best lambda gained from this fine search is 0.00197229 or $10^{-2.71}$, which achieves accuracy almost 54% on validation datasets.

Table 4. The best lambda reached from coarse-to-fine random search

Search	Iteration	Range of lambda	Top three lambda
Coarse	0	$[10^{-5}, 10^{-1}]$	[0.0000316227766, 0.00316227766, 0.001]
Fine	1	$[10^{-4.5}, 10^{-2.5}]$	[0.00116757, 0.00115688, 0.00269959]
	2	$[10^{-2.94}, 10^{-2.57}]$	[0.00158611, 0.00189104, 0.00201792]
	3	$[10^{-2.8}, 10^{-2.7}]$	[0.00197229, 0.00189488, 0.00185231]



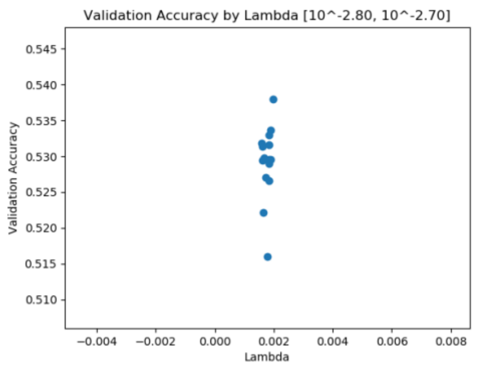


Figure 4. Fine search of the best lambda in the first iteration (top left), second iteration (top right) and third iteration (bottom middle)

2.3. More training samples

In the last experiment, using almost all data for training and leave the last 1,000 samples for validation, lambda 0.00197229, batch size 100, step size 1,000, 12 epoch, and approximately 3 cycles, I got 59.11% accuracy on training data, 52.50% accuracy on validation data, and 51.51% accuracy on testing data. Figure 5 shows that adding more data could improve the generalization power of the model as the gap between training accuracy (blue line) and validation accuracy (orange line) becomes smaller.

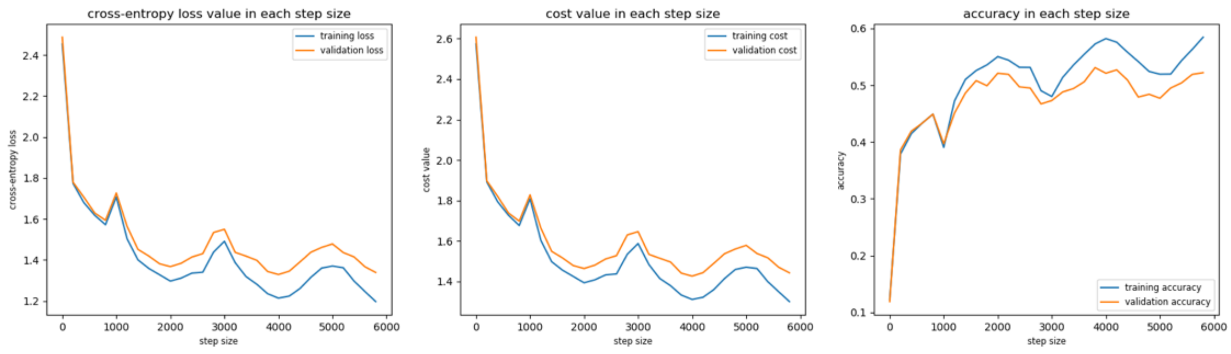


Figure 5. Cross-entropy loss (left), cost value (middle), and accuracy (right) as a function of the step size from the final model

3. Final remarks

From this assignment, it could be learned that applying a cyclical learning rate could improve the accuracy along with a cost of generalization reduction. It is suggested to set the most suitable number of cycles and end up the learning at the end of the cycle when the learning rate reaches the minimal value. However, to deal with the increase of the regularization, one could find out the best lambda value through coarse-to-fine random search. After getting the best lambda value, the model is trained with more datasets. It turns out that apart from the accuracy improvement (51.51%), the regularization capability is far better than at the beginning of the experiment.