Deep Learning in Data Science, DD2424

Short report on lab assignment 2

Classifier

Anna Sánchez Espunyes

April 12, 2021

1 Main objectives and scope of the assignment

In this assignment you will train and test a two layer network with multiple outputs to classify images from the CIFAR-10 dataset. You will train the network using mini-batch gradient descent applied to a cost function that computes the cross-entropy loss of the classifier applied to the labelled training data and an L2 regularization term on the weight matrix.

2 Methods

In this assignment we worked using Python and some of its well-known libraries, including matplotlib, numpy. We have also used the PyCharm IDE and Github.

3 Results and discussion

3.1 Gradient calculation

Gradient calculation is a key step when updating the weights of both layers. In the assignment the gradients are calculated analytically but an extra numerical function is provided to check if the calculations are precise. After checking whether the calculations are correct, it can be stated that the computations are accurate. I have compared the numerically and analytically computed gradient by examining their absolute differences. The absolute differences is in the order of 1e-8, which is a threshold small enough to consider the calculations valid. The maximum absolute difference for each weight matrix and bias vector is the following

Gradient W1	8.105e-09
Gradient W2	1.156e-09
Gradient B1	4.651e-09
Gradient B2	4.546e-08

3.2 Loss, cost and accuracy curves

It is very difficult to choose the appropriate learning rate. If the learning rate is too big then it may lead to divergences. On the other hand a small learning rate leads to very slow training. In this section a solution proposed by Smith, 2015 called cyclical learning rate will be implemented. The idea behind this implementation is to change the learning rate from small values to big values and then to small values periodically, where every period is called cycle (see Figure 1)

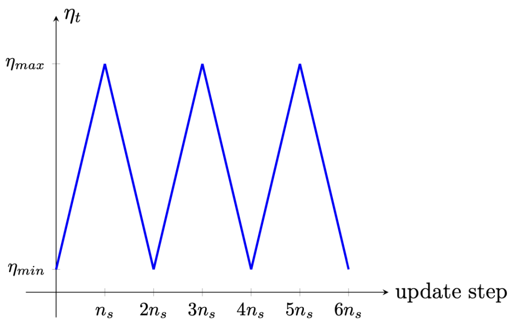


Figure 1: Cyclic learning rate

We need to define a minimum and a maximum learning rate. In the first exercise we have defined the minimum learning rate to 1e-5 and the maximum to 1e-1, lambda=0.01 and the steep size to 500. We will only train for a full cycle, which means that we will train for 10 epochs.

$$epochs/cycle = \frac{2 * stepsize}{numberofbatches} \tag{1}$$

The results obtained implementing 1 cycle of cyclical learning rate can be seen in Figure 2. In the cost and loss figures it is observed that the plot is almost monotonously decreasing and we achieve high accuracy in the validation set. All these properties are achieved thanks to the exploring capacity and getting close to the minima of the cyclic learning rate technique. The accuracy using the test set is 45.9%, which is really close to the guidelines.

In the next exercise we have implemented three cycles (see the results in Figure 3). It can be seen that there are three peaks in the cost and in the loss curves. These peaks correspond to when the learning rate is maximum whereas we get minimum valleys when the learning rate is minimum. This technique ensures that we have the suitable balance between moving to the minimum and exploring. The accuracy in the test set is approximately 46.45%. The difference between this accuracy and the guidelines may be due to the randomness in the initialization.

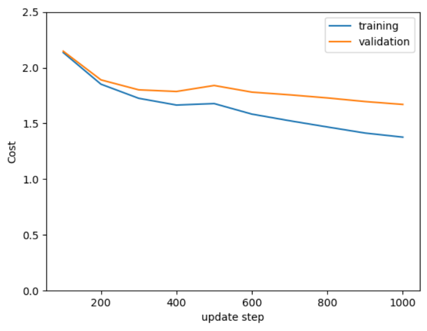
3.3 Regularization optimization

3.3.1 Coarse search

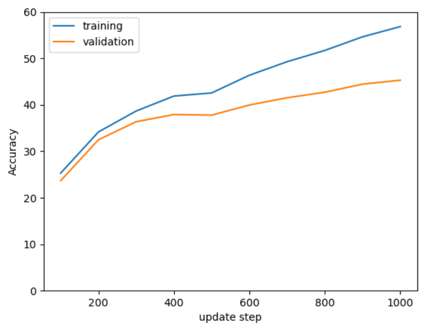
A wide exploration of the regularization term will be performed in order to find a narrower range. I have chosen 10 values between 10e-6 and 10e-1 drawn from a uniform distribution. I have also used most of the training data available, leaving only 5000 images for validation. When training each network, I have used 2 cycles of training. The hyper-parameters settings for the top 3 performing networks are: size of the batch = 100, number of epochs = 18, minimum learning rate = 1e-5, maximum learning rate = 1e-1, step size = 450. The regularization term with the corresponding validation accuracy can be found in Table 1.

3.3.2 Fine search

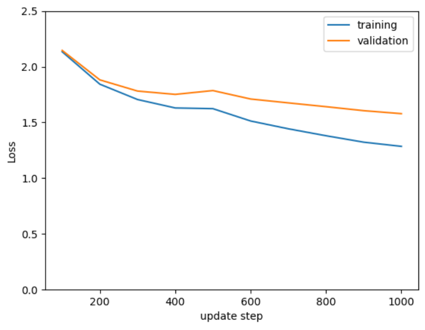
Now that we have detected the most promising regularization terms, we can do a more narrow search. The range will now be between 0.0001 and 0.0009, using 2 cycles for training and the following hyper-



(a) Cost plot



(b) Accuracy



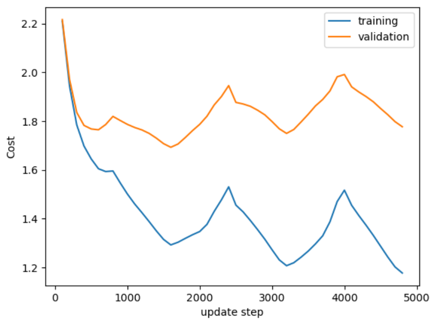
(c) Loss

Figure 2: Parameter settings: eta min = 1e-5, eta max = 1e-1, lambda=.01 and n s=500. As the batch size is 100, one full cycle corresponds to 10 epochs of training.

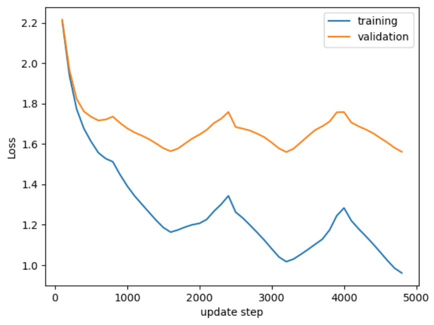
parameters: size of the batch = 90, number of epochs/cycle = 4, minimum learning rate = 1e-5, maximum learning rate = 1e-1, step size = 1000. The regularization term with the corresponding validation accuracy can be found in Table 2.

3.3.3 Best classifier

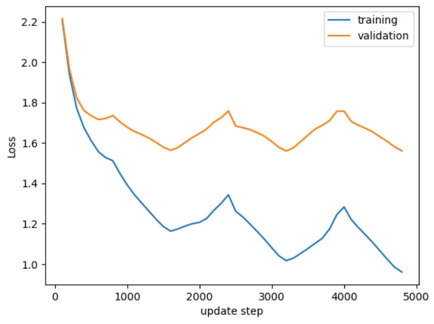
I will train the final model with more data (only using 1000 images for validation instead of 5000) and using the best regularization term found in the narrow search (0.0008). The parameters are the same ones as in the previous exercise but using three cycles. The loss for the training and the test data can be seen in Figure 4. The accuracy in the test set is 51.47, which is worse than in the previous exercise.



(a) Cost



(b) Accuracy



(c) Loss

Figure 3: Parameter settings: eta min = 1e-5, eta max = 1e-1, lambda=.01 and n s=800. With the number of epochs in 50 we will get 3 cycles.

Lambda	Validation accuracy
0.000549	45.36
0.000103	**45.84**
0.03733	40.38
1.558e-06	45.66
0.000302	45.58
0.000658	44.6
6.114e-05	**45.96**
2.299e-06	**45.74**
0.000793	45.66
0.08546	35.22

Table 1: Validation accuracy for different regularization terms

Lambda	Validation accuracy
0.0002	51.8
0.0003	51.7
0.0004	51.34
0.0005	52.2
0.0006	**52.2**
0.0007	51.55
0.0008	**52.26**
0.0009	**52.56**

Table 2: Validation accuracy for different regularization terms

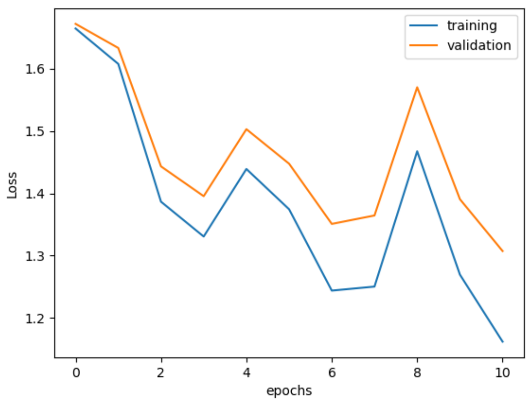


Figure 4: Loss for training and validation