Deep Learning in Data Science, DD2424

Short report on lab assignment 1

Bonus part

Anna Sánchez Espunyes

March 26, 2021

1 Main objectives and scope of the assignment

In this assignment you will train and test a one layer network with multi-ple 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 Exercise 2.1: Optimize the performance of the network

From the proposed optimizations I have done the following three improvements:

1. Use all available training data for training (all five batches minus a small subset of the training images for a validation set and decreasing the size of the validation set.
2. Shuffle the order of your training examples at the beginning of every epoch
3. Decaying the learning rate by a factor 0.9 after each epoch.

For each one of the improvements I will discuss why it is a good technique and I will compare the accuracy obtained when using the improvement and the accuracy without any improvement.

1. **More data**. It is obvious that the more training data is used, the better the algorithm can understand the underlying function. The accuracy obtained without any improvement is 39.02 whereas the accuracy obtained using more data is 41.03. (The parameters used to get this results are the same ones as in the experiment 3 in the previous exercise).
2. **Shuffling**. By shuffling we make sure that we do not train on the same input. This characteristic makes it difficult to get stuck in local minima. The accuracy obtained without any improvement is 39.02 whereas the accuracy obtained using shuffling is 39.17.

3. **Decaying the learning rate**. This improvement makes us explore more in the beginning of the algorithm and make smaller steps as the epochs go by. In this improvement we will start in a higher learning rate (0.01) and multiply the learning rate per 0.9 in every epoch. The accuracy obtained without any improvement is 39.02 whereas the accuracy obtained using shuffling is 39.72

The improvement that brought the largest improvement is using more training data. By using the following parameters, we get the highest accuracy reported.

batch size = 80, eta intial = 0.01, number of epochs =40, lambda=0. With this parameters the test accuracy is 41.36

3.2 Exercise 2.2: Train network by minimizing the SVM multi-class loss

The second exercise is to train a network by minimizing the SVM multi-class loss instead of the cross-entropy loss. The loss will look as follows

$$L_i = \sum_{j \neq y_i} \left[\max\left(0, w_j^T x_i - w_{y_i}^T x_i + 1\right) \right] \tag{1}$$

The calculation of the gradients will also change, and it is as follows for the correct class

$$\nabla_{w_{y_i}} L_i = -\left(\sum_{j \neq y_i} \mathbb{1}\left(w_j^T x_i - w_{y_i}^T x_i + \Delta > 0\right) \right) x_i \tag{2}$$

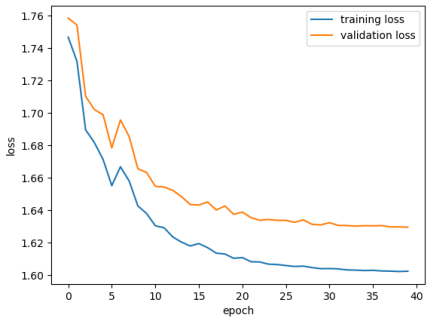
If it is not from the correct class $j \neq y_i$ then

$$\nabla_{w_j} L_i = \mathbb{1}\left(w_j^T x_i - w_{y_i}^T x_i + \Delta > 0\right) x_i \tag{3}$$

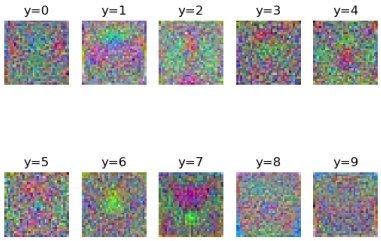
Now we will compare the test accuracy for different settings training with SVM and cross-entropy.

- batch size = 80, eta intial = 0.01, number of epochs =40, lambda=0 (Figure 1). The test error for cross-entropy is 41.36 whereas the test error for SVM is 36.84
- batch size = 1000, eta intial = 0.001, number of epochs =40, lambda=0 (Figure 2). The test error for cross-entropy is 36.24 whereas the test error for SVM is 39.18
- batch size = 80, eta intial = 0.01, number of epochs =40, lambda=1 (Figure 3). The test error for cross-entropy is 38.27 whereas the test error for SVM is 39.22
- batch size = 80, eta intial = 0.01, number of epochs =10, lambda=0 (Figure 4). The test error for cross-entropy is 41.16 whereas the test error for SVM is 35.93

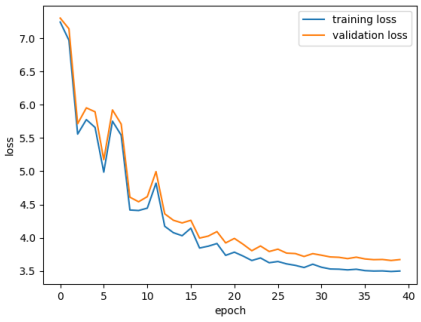
From these experiments we can extract that depending on the hyper-parameters SVM or cross-entropy loss gives better accuracy.



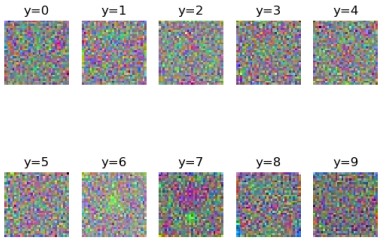
(a) (Loss for cross-entropy



(b) Weights for cross-entropy

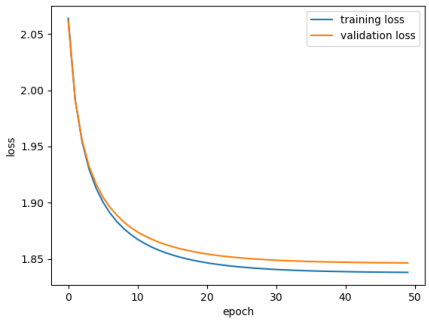


(c) Loss for SVM

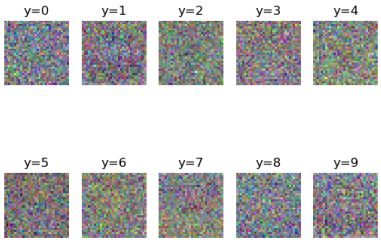


(d) Weights for SVM

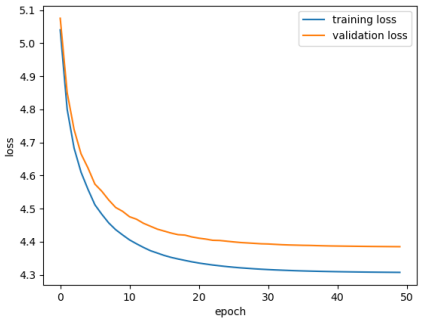
Figure 1: Parameters: batch size = 80, eta intial = 0.01, number of epochs =40, lambda=0.



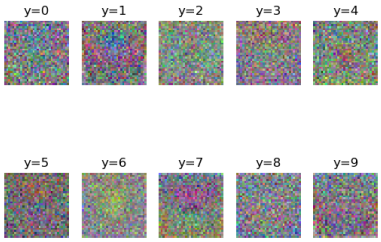
(a) (Loss for cross-entropy



(b) Weights for cross-entropy

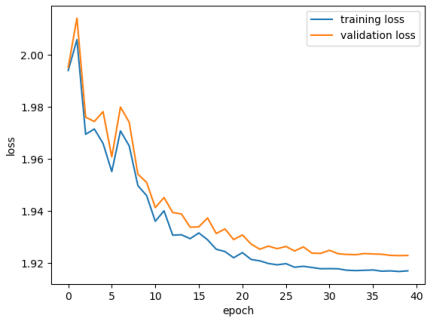


(c) Loss for SVM

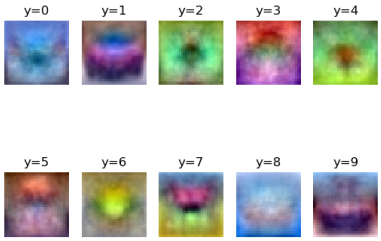


(d) Weights for SVM

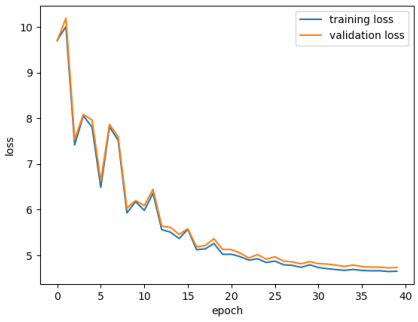
Figure 2: Parameters: batch size = 1000, eta intial = 0.001, number of epochs =40, lambda=0



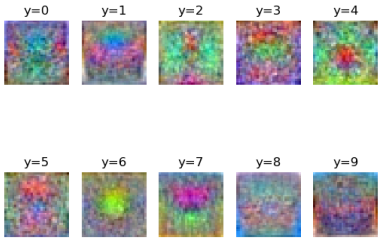
(a) (Loss for cross-entropy



(b) Weights for cross-entropy

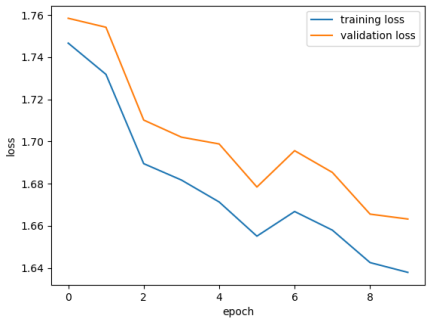


(c) Loss for SVM

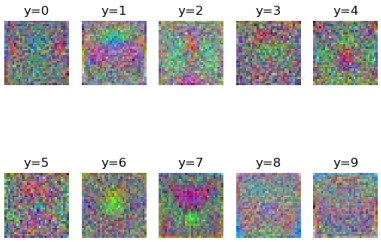


(d) Weights for SVM

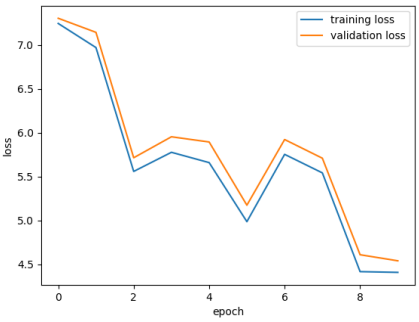
Figure 3: Parameters: batch size = 80, eta intial = 0.01, number of epochs =40, lambda=1



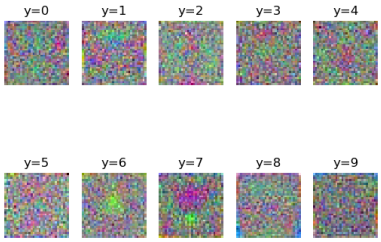
(a) (Loss for cross-entropy



(b) Weights for cross-entropy



(c) Loss for SVM



(d) Weights for SVM

Figure 4: Parameters: batch size = 80, eta intial = 0.01, number of epochs =10, lambda=0