Deep Learning lab 1

Marcus Loberg

April 2023

Introduction

In this assignment a one layer network is used to classify images from the CIFAR-10 dataset. The network will be trained by using mini-batch gradient descent applied to a cost function that computes (firstly) the cross-entropy loss of the classifier applied to the labelled training data and an L2 regularization term on the weight matrix. Later multiple binary cross-entropy loss will be instead used.

Figure 1: CIFAR-10$_examples$

1 Checking gradients against numerical gradients

By calculating the gradients numerical by the provided python code which uses the (finite difference method) we can calculate the error by subtracting by our gradient. This is the result first for the $W[0]$ and than b. A stepsize oh $h = 0.001$ was used

Error $W[0]$:

$$\begin{bmatrix} \text{-7.44e-06} & \text{-2.83e-07} & \text{-2.46e-05} & \vdots & \text{-4.18e-05} & \text{-4.18e-05} & \text{-3.89e-05} \end{bmatrix}$$

Error b :

$$\begin{bmatrix} \text{-1.10e-04} & \text{-1.20e-04} & \text{-1.25e-04} & \vdots & \text{-1.25e-04} & \text{-1.25e-04} & \text{-1.17e-04} \end{bmatrix}$$

2 Exercise 1 (without improvements)

2.1 lambda=0, n_{epochs}=40, n_{batch}=100, eta=.1

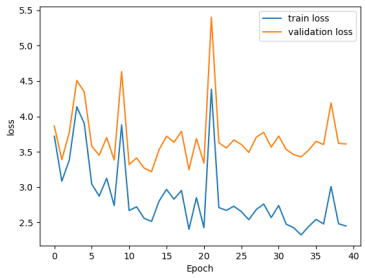

Figure 2: Loss

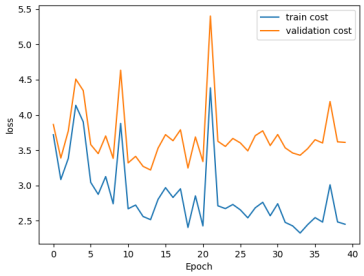

Figure 3: Cost

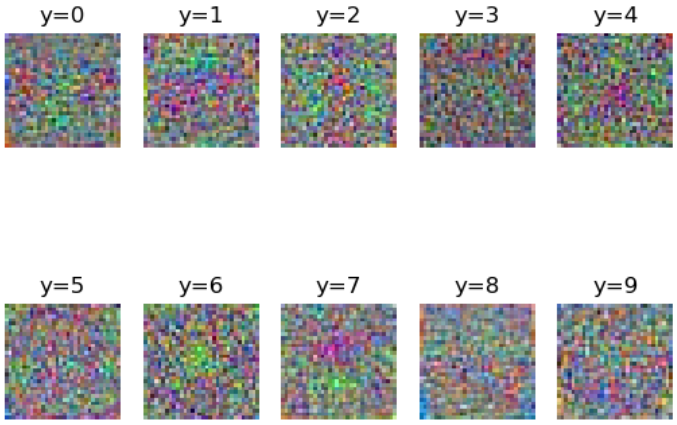

Figure 4: learnt weight matrix

final acc: 0.2827

2.2 lambda=0, n_{epochs}=40, n_{batch}=100, eta=0.001

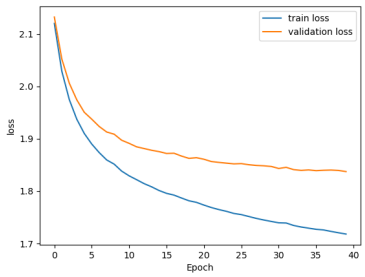

Figure 5: Loss

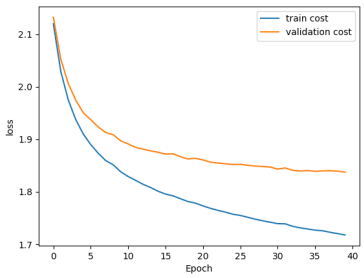

Figure 6: Cost

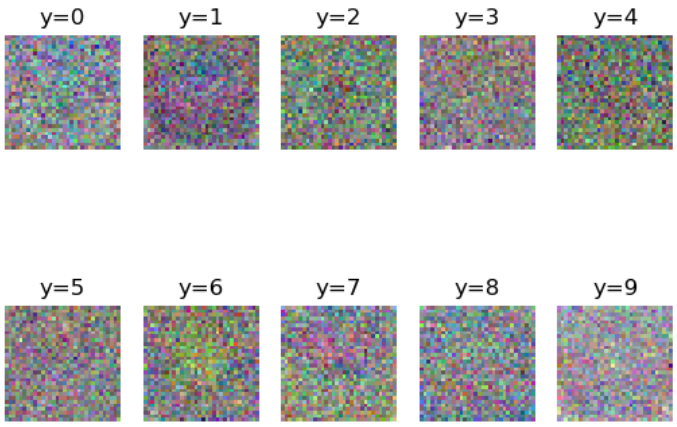

Figure 7: learnt weight matrix

Final acc = 0.3653

2.3 lambda=.1, n_{epochs}=40, n batch=100, eta=.001

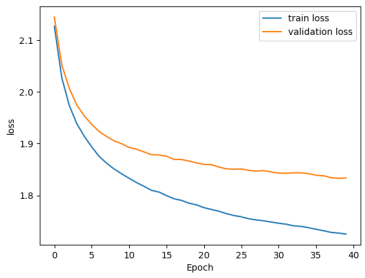

Figure 8: Loss

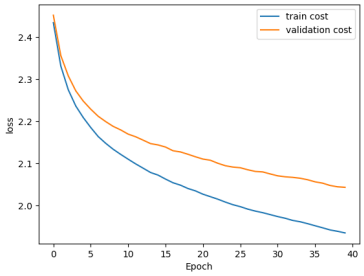

Figure 9: Cost

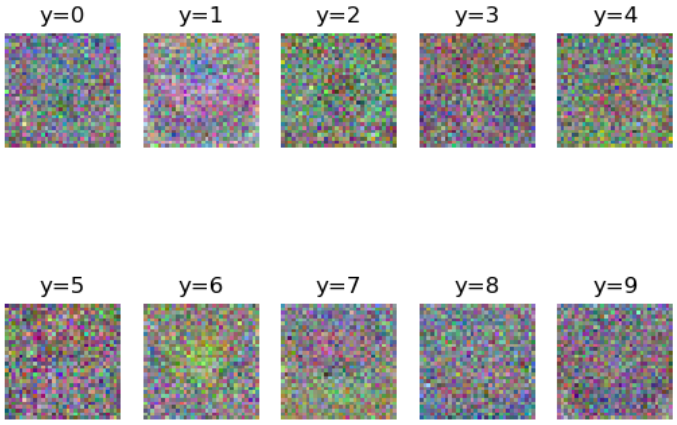

Figure 10: learnt weight matrix

Validation accuracy: 0.3675

2.4 lambda=1, n_{epochs}=40, n_{batch}=100, eta=.001

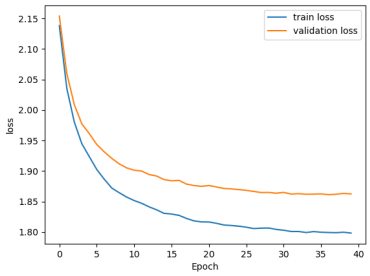

Figure 11: Loss

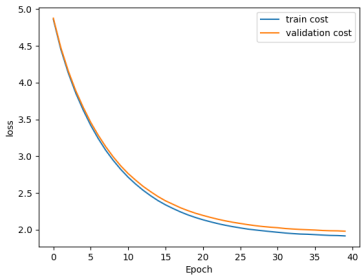

Figure 12: Cost

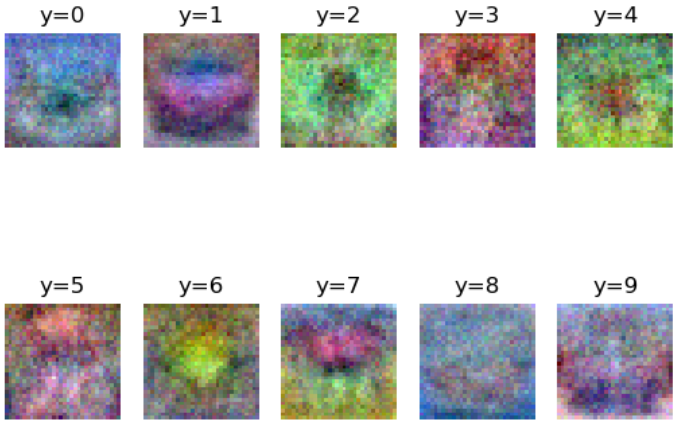

Figure 13: learnt weight matrix

Validation accuracy: 0.3646

2.5 Comment on regularization and learning rate

Increasing Lambda effects the cost of having large weights, this makes so the gradient descent will tend to not specialize as much on the training set, which should lead to better regualization over the validation data. This can be seen

in the Cost plots of the last three tests, higher Lambda leads to a cost which is more alike across training and validation. This also can be seen in the visualized weights, where a higher eta lead to weights which regulized better to look like the images for that label.

The learning rate Eta effects how much the Weights and Biases will be effected for every new batch of gradient descent calculated. A too low Eta will will lead to the algoritm to not learn as it jumps over the optimal place.

3 Exercise 2 (With improvements)

Improvements implemented:

- Use all the available training data for training
- Augment training data by horizontally flipping all and adding to dataset, doubling amount of training data
- Implemented gridsearch for n_{batch}, Eta , n_{epochs} , $lambda$ and $Alpha$
- decaying the learning rate by a factor $Alpha$ every episode

3.1 cross-entropy loss, Found through gridsearch: lambda=0.1, n_{batch}=25, eta=.003,alpha = 0.97

Besides the parameters found by gridsearch, the number of epochs was locked to 100 to easily be able to compare to the other tests.

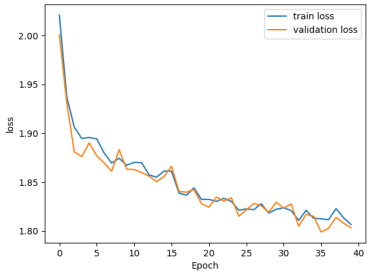

Figure 14: Loss

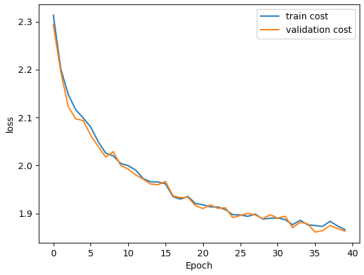

Figure 15: Cost

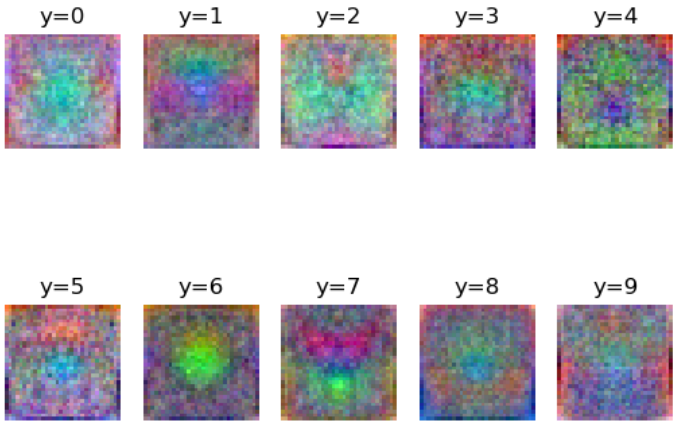

Figure 16: learnt weight matrix

Validation accuracy = 0.401

4 Exercise 3 (multiple binary cross-entropy loss)

$$\frac{\partial L_{\text{multiple BCE}}}{\partial s} = \frac{\partial L_{\text{multiple BCE}}}{\partial p} \frac{\partial p}{\partial s} \tag{2}$$

We now insert:

$$P = \sigma(s) = \frac{e^s}{e^s + 1} \tag{3}$$

and

$$L_{\text{multiple BCE}} = -y * log(p) + (1 - y) * log(1 - p) \tag{4}$$

which results in:

$$\frac{\partial L_{\text{multiple BCE}}}{\partial s} = \frac{p - y}{p(1 - p)} p(1 - p) = p - y \tag{5}$$

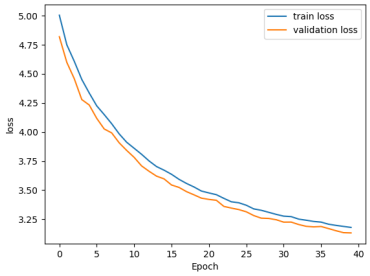

Figure 17: Loss

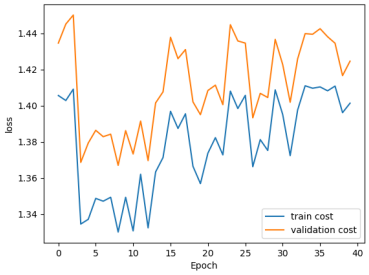

Figure 18: Cost

4.1 multiple binary cross-entropy loss, lambda=0.1, n_{batch}=25, eta=.003,alpha = 0.97

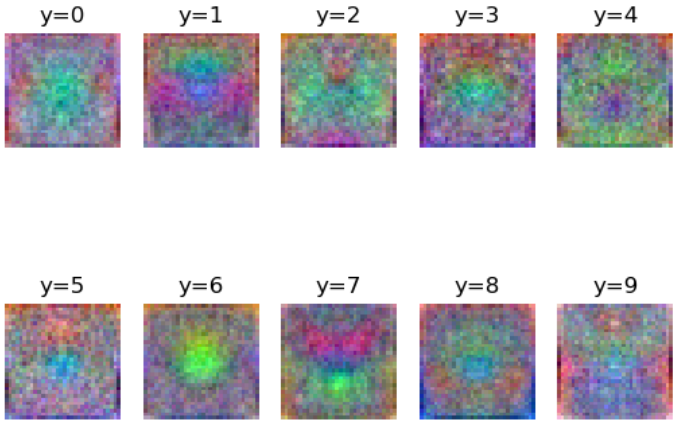

Figure 19: learnt weight matrix

Validation accuracy = 0.427

The improvement that improved the training the most was changing the evaluation to use the sigma function and the multiple binary cross-entropy loss instead of cross validation implemented. Comparing the validation accuracy of the network trained with the multiple binary cross-entropy loss compared to the

cross-entropy loss shows that the accuracy imroved from 0.401 to 0.427.

Looking at the loss and cost plots (figure 14/15 and Figure 17/18) we can see that the multiple binary cross-entropy loss gave a loss that is much the same as the loss in the cross validation loss but the validation and test are a bit further away from eachother in the multiple binary cross-entropy loss, especially for the cost. This should mean that there is more overfitting for the multiple binary cross-entropy loss.

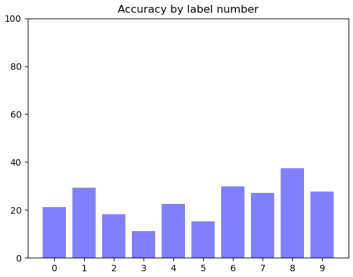

Figure 20: softmax + cross-entropy

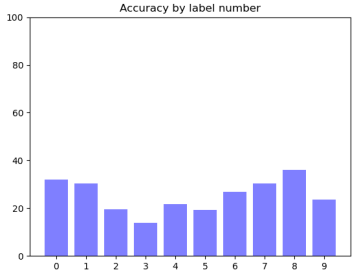

Figure 21: Sigmoid + multiple binary cross-entropy losses

These barplots show the the accuracy for each label individually for the two different methods used in this assignment. Some were uneffected like 9(Truck) and 4(deer), but 0 (Planes) saw a hude improvement, going from 20 to above 30.