Assignment 1

Albin Larsson Forsberg

April 17, 2020

1 Introduction

The assignment consists of creating a basic one layer network that is classifying images from the CIFAR-10 dataset. The inputs consists of a input layer of size 3072 and the output layer is a softmax function. Whichever node has the highest probability assigned to it will be the prediction of the network. Using this network set up an accuracy of 38.48 percent on test data was achieved with a model with no regularization, and a model with regularization achieved an accuracy of 39.21 on the test data.

2 Dataset

The dataset used is the CIFAR-10 dataset that is coming from the Canadian Institute For Advanced Research. It contains in total 60,000 picture from 10 categories: airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. It is widely used in Machine Learning to develop and train different classification models.

3 Model

The Model used is the most basic model with only two layers, an input and an output layer. The input consists of the rgb data from each pixel and the three dimensional Matrix structure has been vectorized to only be in one dimension. This vector is the fed into the network that predicts the classifier. Each node is described by function 1. Where W are the weights and b the biases.

$$f(X) = \text{softmax}(WX + b) \tag{1}$$

The model was trained using a version of gradient descent which is called mini-batch gradient descent. It is almost like stochastic gradient descent, but instead of using individual train samples as a training step, it uses a collection of training samples. In this report the size of the batches are 100 samples each. The model variables were initialized using a normal distribution with $\sigma^2 = 0.01$.

η	λ	Acc_{train}	Acc_{test}
0.1	0	45.85 %	29.48 %
0.001	0	44.3 %	38.48 %
0.001	0.1	44.45 %	39.21 %
0.001	1	39.88 %	37.55 %

Table 1: Results from 4 runs with different hyper parameters.

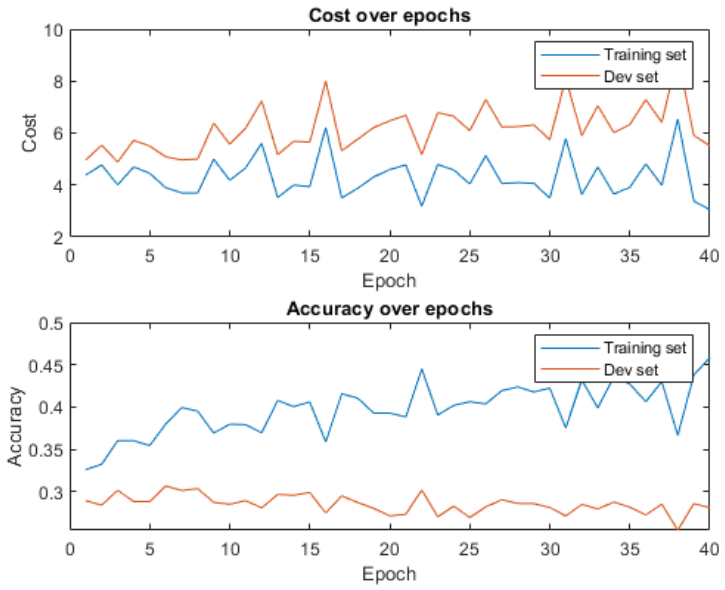

Figure 1: Loss over time for run 1

The gradient was evaluated analytically and comparing the results with a numerical approximation of the gradient showed that the evaluation was correct. Using a relative error as a comparison, the error was in the order of magnitude 10^{-7}, which was deemed adequate. This can be seen in row 85 of the submitted code.

4 Results

The model worked and was implemented correctly, the results from four different runs is seen in table 1

4.1 Run 1

Parameters for run 1 was $\eta = 0.1$, $\lambda = 0$, $N_{batch} = 100$, and $N_{epochs} = 40$.

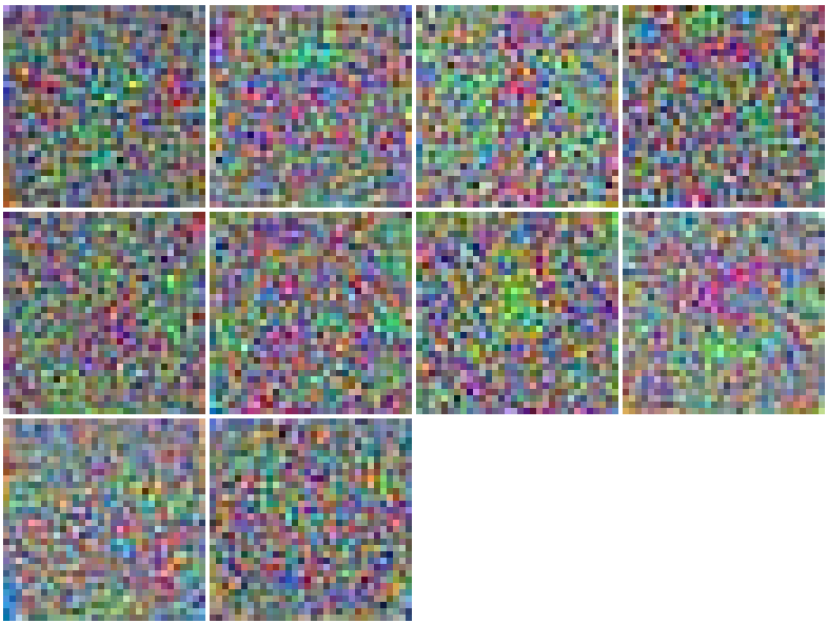

Figure 2: Visualized model for the ten classifiers for run 1

4.2 Run 2

Parameters for run 2 was $\eta = 0.001$, $\lambda = 0$, $N_{batch} = 100$, and $N_{epochs} = 40$.

4.3 Run 3

Parameters for run 1 was $\eta = 0.001$, $\lambda = 0.1$, $N_{batch} = 100$, and $N_{epochs} = 40$.

4.4 Run 4

Parameters for run 1 was $\eta = 0.001$, $\lambda = 1$, $N_{batch} = 100$, and $N_{epochs} = 40$.

5 Conclusions

Increasing the regularization means that the models capacity to fit to the variance in the training data is reduced, and will reduce the flexibility of the model. However, if the training data is not a perfect example of the distribution of the true underlying source, then the regularization could help with fitting the model in such a way that it generalizes better. We can see that different in a very clear way when we run the second and fourth scenario. The difference in hyper parameters is only the regularization term λ, and the difference between the two fits is striking. It can clearly be seen that the model with some regularization

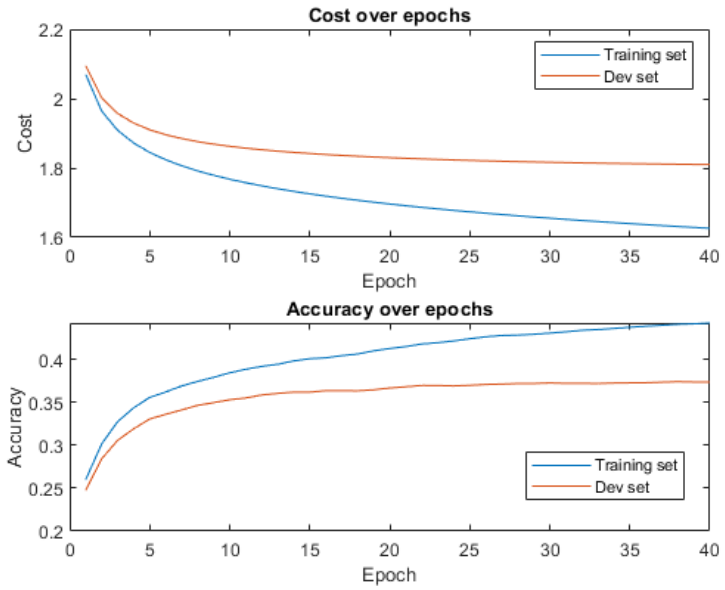

Figure 3: Loss over time for run 2

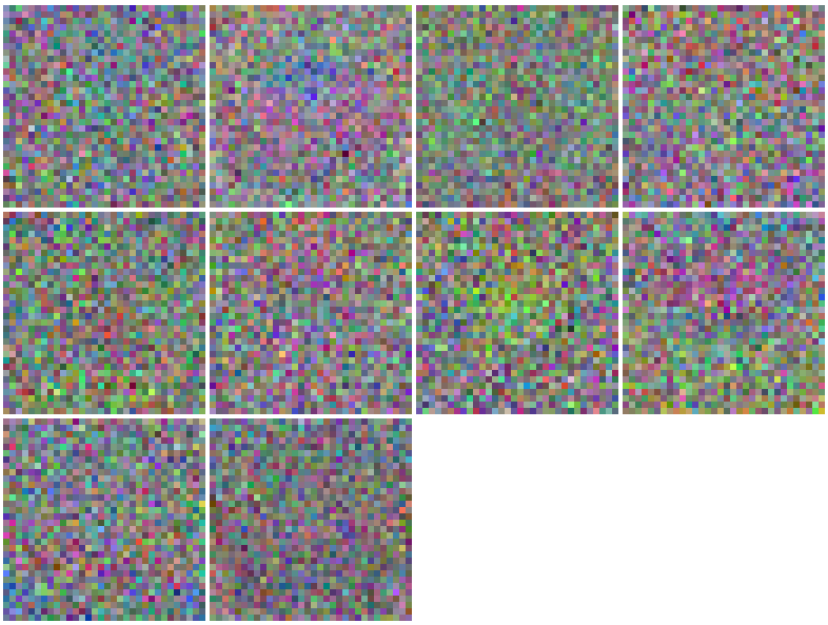

Figure 4: Visualized model for the ten classifiers for run 2

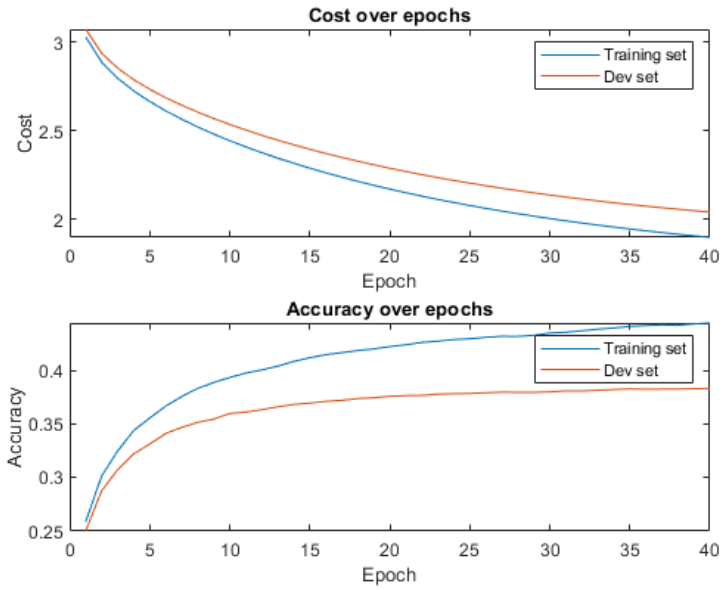

Figure 5: Loss over time for run 3

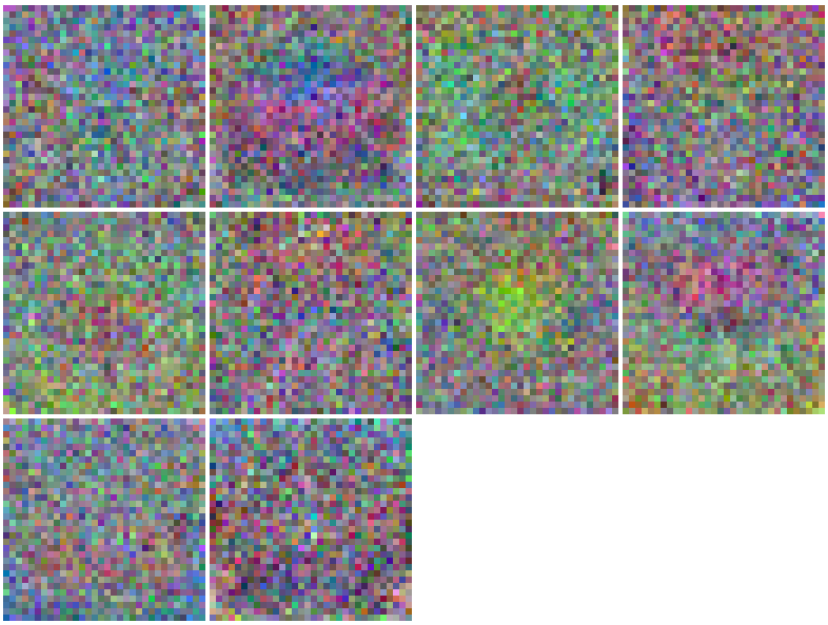

Figure 6: Visualized model for the ten classifiers for run 3

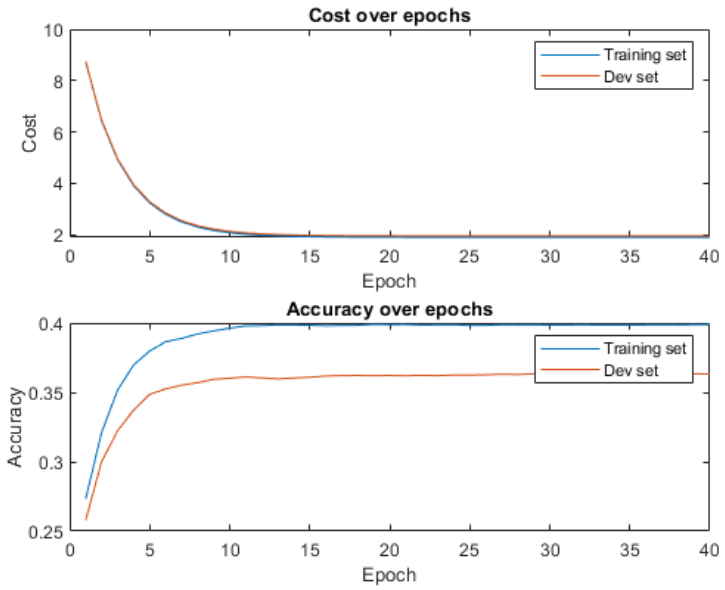

Figure 7: Loss over time for run 4

Figure 8: Visualized model for the ten classifiers for run 4

fits worse to the training set, but that if the regularization is selected correctly it will perform better as it will not overfit to the data.

Finding the right learning rate is important. Even if we get a better result when using a bigger learning rate, it is clear to see that the loss development does not always get better, it sometimes takes a step too big in the direction of the gradient. This however means that it will learn fast, but also run the risk of never finding a true optimum.