Assignment 1 Bonus Points - DD2424

August Regnell 970712-9491

6 April, 2021

Contents

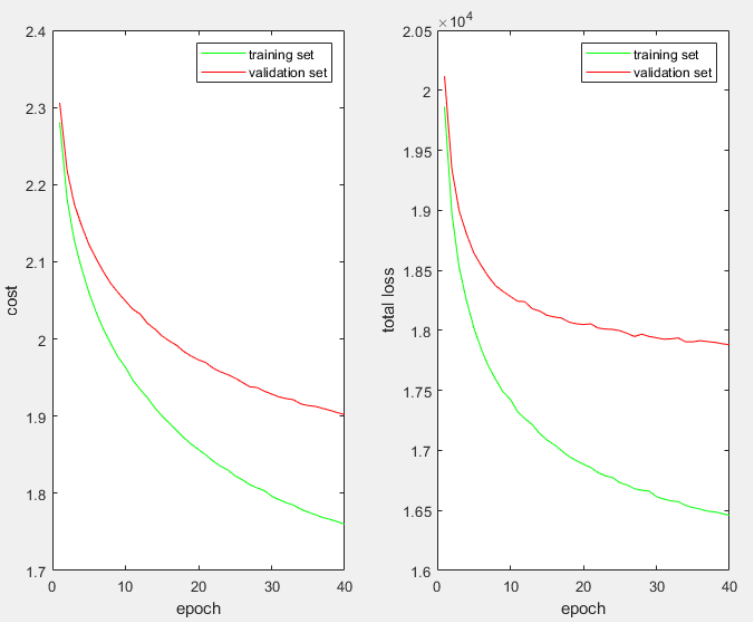
1 Exercise 2.1

I will use the best performing settings from the original assignment as a benchmark. That is,

n_batch	n_epochs	η	λ
100	40	0.001	0.1

Figure 1: Training parameters for the benchmark

which gave the following results



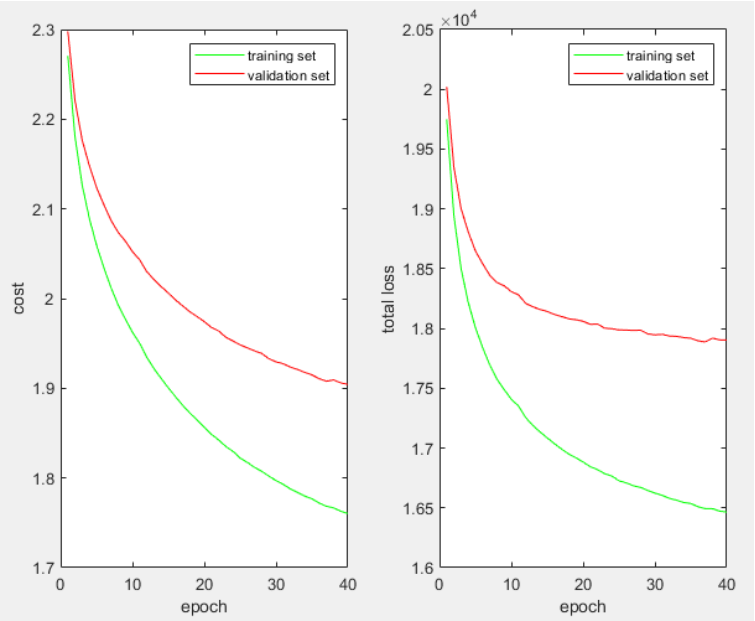
(a) Total loss and the cost function of the benchmark

Training accuracy	Test accuracy
44.76%	39.03%

(b) Accuracies of benchmark

1.1 Improvement 1 - Shuffled order of training examples

At the beginning of every epoch the training examples are shuffled. Using the settings stated in Figure 1 we got the following results



(a) Total loss and the cost function of the benchmark

	Training	Test
Accuracy	44.72%	39.41%
$\Delta_{accuracy}$	-0.04%	+0.38%

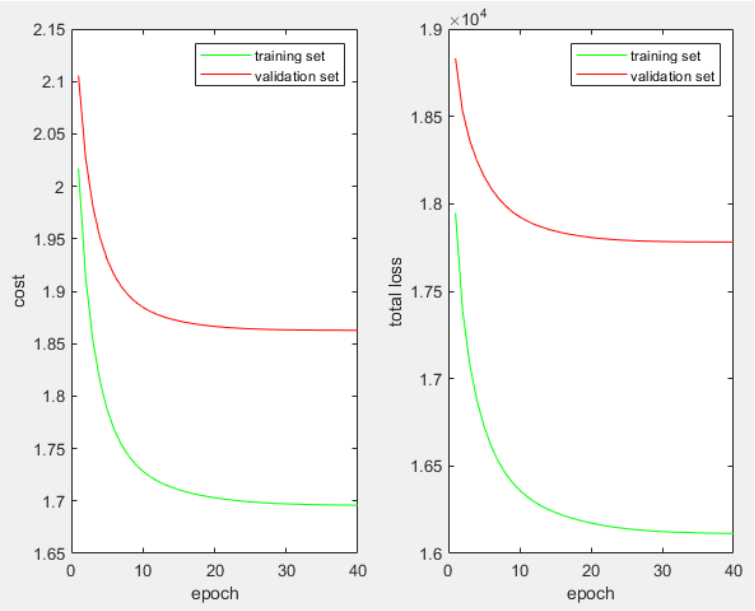
(b) Accuracies of shuffling

1.2 Improvement 2 - Decaying learning rate

At the end of every epoch the learning rate will decrease with a factor μ. That is $\eta_{t+1} = \mu \cdot \eta_t$. We can now start with a higher learning rate. The following (starting) settings were used

n_batch	n_epochs	η	λ
100	40	0.01	0.1

Figure 4: Training parameters for the decaying learning rate



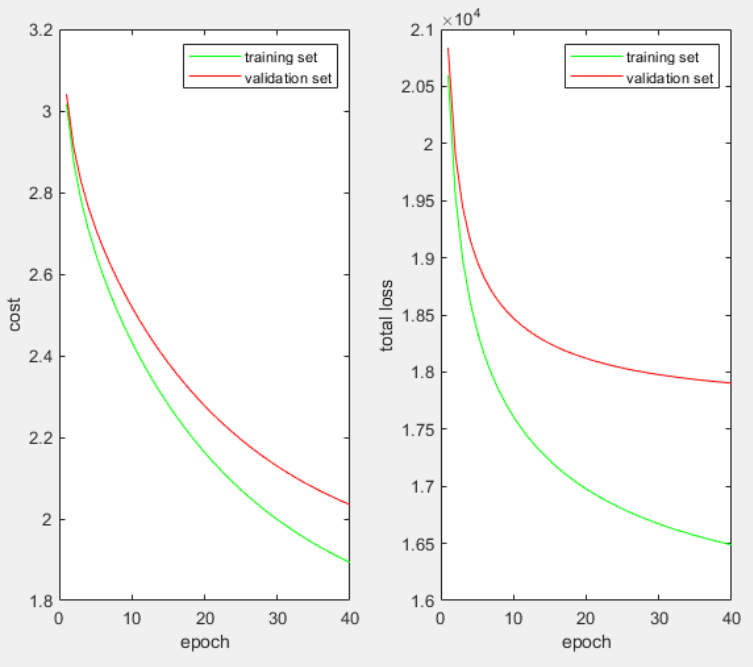
(a) Total loss and the cost function of the decaying learning rate with $\mu = 0.9$

μ	Test accuracy	$\Delta_{testaccuracy}$
0.95	39.26%	+0.23%
0.90	39.64%	+0.61%
0.85	39.52%	+0.49%
0.80	39.50%	+0.47%

(b) Accuracies of decaying η vs benchmark

1.3 Improvement 3 - Xavier initialization

Now the weigths are initialized as follows $W_{i,lm} \sim N(w; 0, \sigma^2)$ where $\sigma = \frac{1}{\sqrt{n_{in}}}$ and W_i has size $n_{out} \times n_{in}$. In our case this gives $\sigma = \frac{1}{\sqrt{3072}} \approx 0.018$. Using the settings stated in Figure 1 we got the following results



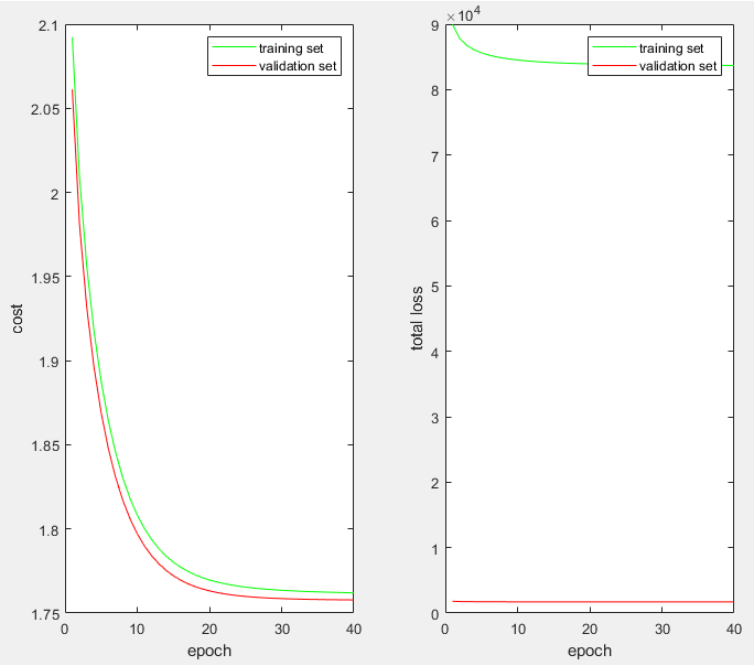
(a) Total loss and the cost function of Xavier inatialization

	Training	Test
Accuracy	44.67%	39.05%
$\Delta_{accuracy}$	-0.09%	+0.02%

(b) Accuracies of the Xaiver initalization

1.4 Improvement 4 - Using all the available training data

We now use all the available training data, 49,000 datapoints for training and 1,000 for validation. Using the settings stated in Figure 1 we got the following results



(a) Total loss and the cost function when using all the data

	Training	Test
Accuracy	42.15%	41.02%
$\Delta_{accuracy}$	-2.61%	+1.99%

(b) Accuracies when using all the data

1.5 Improvement 5 - Combined improvements

As a final improvement we combine all the previous improvements, except the decreasing learning rate, while simultaneously increasing the number of epochs. We thus use the following improvements

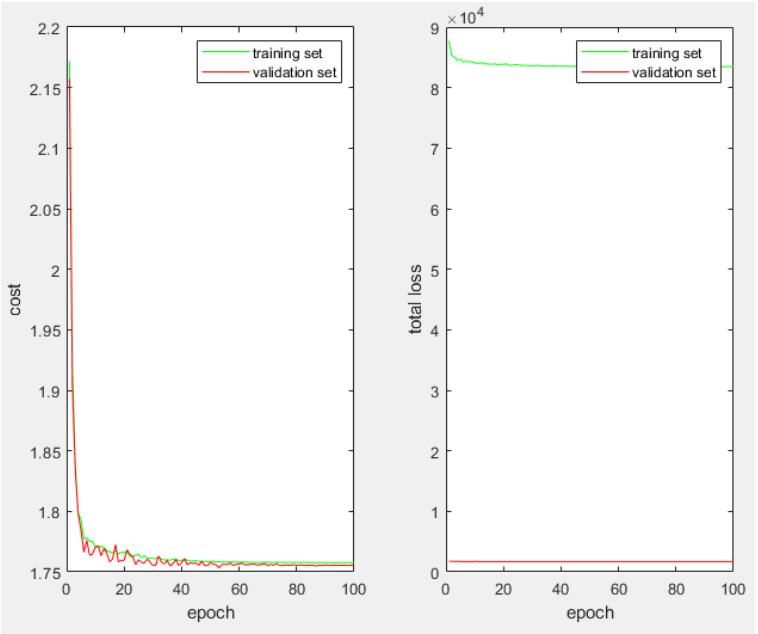
1. Shuffling the training examples at the beginning of every epoch
2. Decrease the learning factor with $\mu = 0.95$ at the end of every epoch
3. Xaiver initialization
4. All the training data
5. Increase the number of epochs

and the following settings

n_batch	n_epochs	η	λ
100	100	0.005	0.1

Figure 8: Training parameters for the combined improvements

which gave the following results



(a) Total loss and the cost function when using all the data

	Training	Test
Accuracy	42.49%	41.07%
$\Delta_{accuracy}$	-2.27%	+2.04%

(b) Accuracies when using all the data

1.6 Summary

We see that using all the available training data gave the individually highest increase in accuracy, followed by using a decreasing learning rate. However, combining all the improvements gave a marginally higher increase in accuracy compared to only using all the available training data. The effects of the combined improvements could most likely be increased by a tuning of the different parameters.

2 Exercise 2.2

For this section we should use the SVM multi-class loss instead of the cross-entropy loss. It is defined as follows for a single data point x_i

$$l_{SVM}(\mathbf{s}, y) = \sum_{j \neq y} max(0, s_j - s_y + 1)$$

where y is the true class. Since $\mathbf{s} = Wx + b$ we get that

$$\frac{\partial l}{\partial w_y} = -\Big(\sum_{j \neq y} I\{s_j - s_y + 1 > 0\} \Big) x_i$$

and

$$\frac{\partial l}{\partial w_j} = I\{s_j - s_y + 1 > 0\} x_i$$

where $I\{\cdot\}$ is the indicator function.
The cost function then becomes

$$L = \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x},y) \in \mathcal{D}} l_{SVM}(\mathbf{x}, y) + \lambda \left\| W \right\|_2$$

The gradient and the cost function defined above was implemented in matlab using matrices to optimize performance.

2.1 Test 1

The following settings were used

n_batch	n_epochs	η	λ
100	40	0.001	0

Figure 10: Settings of test 1

These were the results

	Training accuracy	Test accuracy	$\Delta_{accuracy}$
Cross-entropy loss	45.76%	38.61%	7.15%
SVM loss	48.30%	35.32%	12.98%
$\Delta_{accuracy}$	+2.54%	-3.29%	+5.83 %

Figure 11: Results of test 1

We see a big disparity between the training and test accuracy of the SVM loss, indicating that there may be some overfitting.

2.2 Test 2

Compared to the previous test λ was increased to 0.1.

n_batch	n_epochs	η	λ
100	40	0.001	0.1

Figure 12: Settings of test 2

These were the results

	Training accuracy	Test accuracy	$\Delta_{accuracy}$
Cross-entropy loss	44.69%	39.09%	5.60%
SVM loss	47.98%	36.18%	11.80%
$\Delta_{accuracy}$	+3.29%	-2.91%	6.20%

Figure 13: Results of test 2

The increase of λ decreased the difference between the accuracy on the test and training set, but the decrease was less apparent on the SVM loss.

2.3 Test 3

Compared to the previous test λ was increased to 1.

n_batch	n_epochs	η	λ
100	40	0.001	1

Figure 14: Settings of test 3

These were the results

	Training accuracy	Test accuracy	$\Delta_{accuracy}$
Cross-entropy loss	39.87%	37.49%	2.38%
SVM loss	42.09%	36.18%	5.91%
$\Delta_{accuracy}$	+2.22%	-1.31%	3.53%

Figure 15: Results of test 3

The further increase of λ gave better accuracies for the SVM loss but worse for the cross entropy. This indicates that the SVM loss may need a higher regulation.

2.4 Test 4

λ was kept at 1 and the learning rate was decreased to 0.0001.

n_batch	n_epochs	η	λ
100	40	0.0001	1

Figure 16: Settings of test 4

These were the results

	Training accuracy	Test accuracy	$\Delta_{accuracy}$
Cross-entropy loss	37.65%	36.08%	1.57%
SVM loss	42.10%	37.59%	4.51%
$\Delta_{accuracy}$	+4.45%	+1.51%	2.94%

Figure 17: Results of test 4

We now see that the SVM loss is beating the cross-entropy loss. This is probably due to the gradient of the SVM loss being large, which means that a lower learning rate can more easily handle it. The elements in the gradient takes the values $k \cdot x_{i,j}$ where $k \in \{-(C-1), -(C-2), \ldots, -1, 0, 1\}$ (where C is the number of classes) when excluding regularization, which is comparatively large to the cross-entropy loss. We also see that the low learning rate leads to that the cross-entropy loss is underfitting (it has a small difference between the training and test accuracy).

2.5 Test 5

λ and η were kept at the levels of the previous test but the number of epochs were increased to 100.

n_batch	n_epochs	η	λ
100	100	0.0001	1

Figure 18: Settings of test 5

These were the results

	Training accuracy	Test accuracy	$\Delta_{accuracy}$
Cross-entropy loss	40.12%	37.33%	2.79%
SVM loss	43.64%	37.86%	5.78%
$\Delta_{accuracy}$	+3.52%	+0.53%	2.99%

Figure 19: Results of test 5

Even when we increase the number of epochs we see that the SVM loss still outperforms the cross-entropy loss. This is probably due to that the λ is "fitted" to the SVM-loss. However, we see that the underfitting for the cross-entropy loss has decreased, as it should when the number of epochs are increased.

2.6 Summary

We see that the cross-entropy loss outperforms the SVM loss when using the parameter settings used in the original assignment. To make the SVM loss outperform the cross-entropy loss we need to increase the regularization and deacrease the learning rate. The former is probably due to the SVM loss' apparent tendency to overfit and the latter due to the SVM loss' naturally larger gradients.