# DD2424
# Deep Learning in Data Science
# Assignment 1 Bonus Points

Franco Ruggeri, fruggeri@kth.se

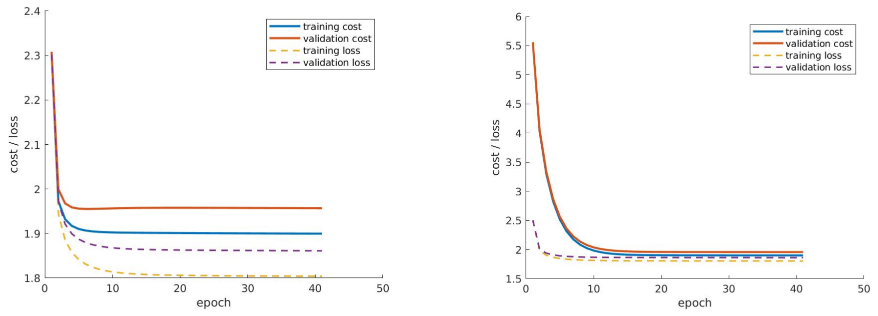
March 30, 2020

## 1 Improvements

I implemented the following improvements:

1. Use of all the available data, with a validation set reduced to 1000 images (100 from each class) (improvement (a) proposed in the instructions).
2. Use of the validation set to train for longer time and avoid overfitting (improvement (b) proposed in the instructions). I implemented this by means of *early stopping*, i.e. when the cost on the validation set increases more than a threshold, the algorithm stops and the best solution so far is taken. The number of updates[1] was fixed to 20000, except for the final test case, where it was set to 100000.
3. Grid search to find good values for $\lambda$, $\eta$ and batch size (improvement (c) proposed in the instructions). In particular, I searched on a total of 300 configurations (5 for the batch size, 3 for $\eta$ and 20 for $\lambda$). With more configurations I could probably improve the result, but running was too time consuming.
4. Decay of the learning rate by a factor of .9 after each epoch (improvement (d) proposed in the instructions).
5. Xavier initialization of weights (improvement (e) proposed in the instructions).

[1]In order to compare fairly, I fixed the number of updates instead of the number of epochs. The number of epochs is then found as ($n_{epochs} = n_{updates}\frac{n_{batch}}{n}$).



(a) (b)

Figure 1: Learning curve with Xavier initialization (a) and with random initialization from $\mathcal{N}(0, 0.01)$ (b) for the fourth parameter setting ($\lambda = 1$, $n_{updates} = 4000$, $n_{batch} = 100$, $\eta = 0.001$).

6. Shuffling the training data before each epoch (improvement (g) proposed in the instructions).

To analyze the benefits systematically, I started by enabling one improvement at a time and tried them on the same parameter settings as in the exercise 1. Then I enabled all, including grid search. The results are reported in table 1.

The largest gain for the first parameter setting is got by decaying the learning rate (improvement 4). This is rather obvious considering that this configuration had too high learning rate and was unstable. A further simple observation is about the drastic decrease with the early stopping: the cost went up and down (instability) but this improvement stops at the first up.

The fourth parameter setting, instead, is almost independent of all these modifications because the regularization forces a very simple model (low variance $\Rightarrow$ improvements 1-2 not so useful). However, it is an example of how Xavier initialization makes training more stable, i.e. the gradients do not vanish and learning is faster (see figure 1).

In general, the most useful enhancement is for sure to increase (by a factor $\approx 5$) the number of training data, which as we know reduces the risk of overfitting.

The best accuracy is 41.09% and corresponds to all the enhancements enabled. As suggested in the instructions, a smaller batch size ($n_{batch} = 26$) gives a better generalization and we can also notice that no regularization is needed, because we increased the amount of training data.

| Setting | $\lambda$ | $n_{updates}$ | $n_{batch}$ | $\eta$ | Improvements | Accuracy (%) |
|---|---|---|---|---|---|---|
| 1 | 0 | 4000 | 100 | 0.1 | - | 29.74 |
| | | | | | 1 | 31.98 |
| | | | | | 4 | 34.43 |
| | | | | | 5 | 29.66 |
| | | | | | 6 | 27.52 |
| | | 20000 | | | 2 | 8.96 |
| 2 | 0 | 4000 | 100 | 0.001 | - | 38.69 |
| | | | | | 1 | 40.42 |
| | | | | | 4 | 37.41 |
| | | | | | 5 | 39.23 |
| | | | | | 6 | 38.95 |
| | | 20000 | | | 2 | 38.75 |
| 3 | 0.1 | 4000 | 100 | 0.001 | - | 39.08 |
| | | | | | 1 | 40.28 |
| | | | | | 4 | 37.49 |
| | | | | | 5 | 39.26 |
| | | | | | 6 | 39.40 |
| | | 20000 | | | 2 | 39.49 |
| 4 | 1 | 4000 | 100 | 0.001 | - | 37.57 |
| | | | | | 1 | 37.16 |
| | | | | | 4 | 37.33 |
| | | | | | 5 | 37.55 |
| | | | | | 6 | 37.65 |
| | | 20000 | | | 2 | 37.82 |
| Grid search | 0 | 100000 | 26 | 0.001 | 1-6 | 41.09 |

Table 1: Accuracy (evaluated on the test set) for several parameter settings and improvements.

## 2 SVM multi-class loss

I derived analytically the gradients for the computational graph with the SVM multi-class loss. The only piece that needs to be modified is $G_{batch}$, whose columns are $\frac{\partial l(\mathbf{x},y,W,\mathbf{b})}{\partial \mathbf{s}}$. So, for one training sample:

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

$$\frac{\partial l}{\partial s_j} = I(s_j - s_y + 1 > 0), j \neq y \tag{2}$$

Table 2 reports the accuracy of networks trained with cross-entropy and SVM multi-class loss, for several test cases. The cross-entropy loss gives better results in all the cases.

| Setting | $\lambda$ | $n_{updates}$ | $n_{batch}$ | $\eta$ | Improvements | Loss function | Accuracy (%) |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 4000 | 100 | 0.1 | - | SVM | 28.98 |
| | | | | | | cross-entropy | 29.74 |
| 2 | 0 | 4000 | 100 | 0.001 | - | SVM | 35.31 |
| | | | | | | cross-entropy | 38.69 |
| 3 | 0.1 | 4000 | 100 | 0.001 | - | SVM | 35.92 |
| | | | | | | cross-entropy | 39.08 |
| 4 | 1 | 4000 | 100 | 0.001 | - | SVM | 36.10 |
| | | | | | | cross-entropy | 37.57 |
| Grid search | 0.4 | 100000 | 100 | 0.001 | 1-6 | SVM | 38.72 |
| | 0 | | 26 | 0.001 | | cross-entropy | 41.09 |

Table 2: Accuracy (evaluated on the test set) with cross-entropy and SVM multi-class loss for several parameter settings.