Henrik Holm henholm@kth.se 2020-03-29

DD2424 Deep Learning in Data Science

Assignment 1 – base version

1. Introduction

The focus of this assignment is on **image classification** with ten possible classes. For this task, a **single-layer neural network** is trained using a "vanilla" implementation of the **mini-batch gradient descent** algorithm. The algorithm is applied to a cost function which computes the **cross-entropy loss** of the classifier applied to labelled training data and an L_2 regularization term on the weight matrix. The data set used is the CIFAR-10 collection of labelled images.

The toolset used for solving the assignment included default, built-in Python as well as the two external Python libraries *numpy* (for linear algebra purposes) and *matplotlib* (for plotting needs).

For making sure that the implemented *compute_gradients* function (which computes gradients *analytically*) yielded the correct values, a test was devised. The test compares the results of *compute_gradients* with the results of *compute_gradients_num*, which calculates the same gradients, however using a numerical approach. The test was conducted using numpy's *allclose*-function, which compares the values of two equally shaped matrices and returns either *True* or *False* depending on whether the compared elements are all close in value or not. With an absolute tolerance of 1e-05, the function returned *True* when comparing gradients computed for , meaning that all elements of the gradients were similar. With an absolute tolerance of 1e-06, the function returned *False.* However, upon close inspection, one could see that apart from a few exceptions, most values were similar. For this reason, the analytical compute_gradients-function implemented was deemed as being correctly implemented.

2. Results

A random seed of "*12345*" was used throughout the assignment. Four different tests were conducted, each with a different set of parameters for the learning algorithm. The parameters were:

- **n_batch**, the size of the mini batches;
- **eta**, the learning rate;
- **n_epochs**, the number of runs through the whole training set;
- **lambda**, the regularization term.

Figures containing plots of results are displayed below. The plots in Figures 1-4 plot the loss function with respect to the number of batches, both for the training and the validation data set. The four different sets of parameters tested were:

1. lambda=0, n_epochs=40, n_batch=100, eta=0.1

   which yielded the following accuracies:

- training set accuracy: 0.4187
- validation set accuracy: 0.2886
- test set accuracy: 0.2811

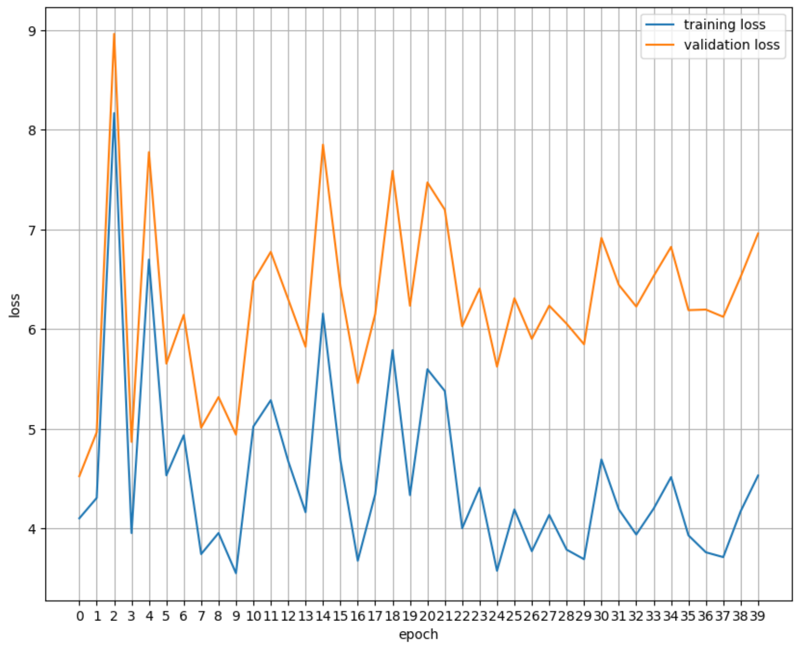

Figure 1: loss function for lambda=0, n_epochs=40, n_batch=100, eta=0.1

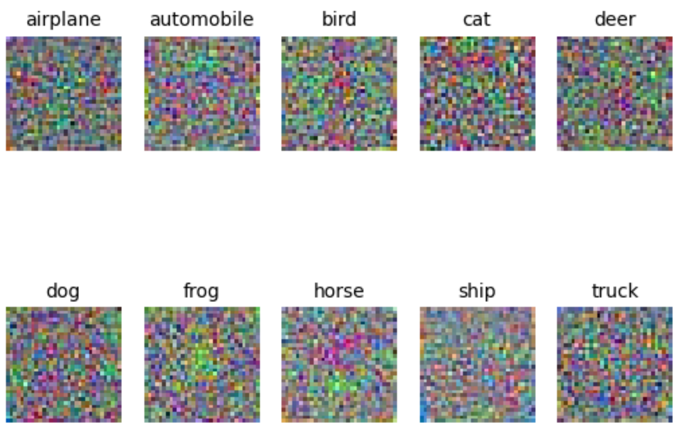

Figure 2: class templates for lambda=0, n_epochs=40, n_batch=100, eta=0.1

2. lambda=0, n_epochs=40, n_batch=100, eta=0.001

   which yielded the following accuracies:

- training set accuracy: 0.4547
- validation set accuracy: 0.3900
- test set accuracy: 0.3860

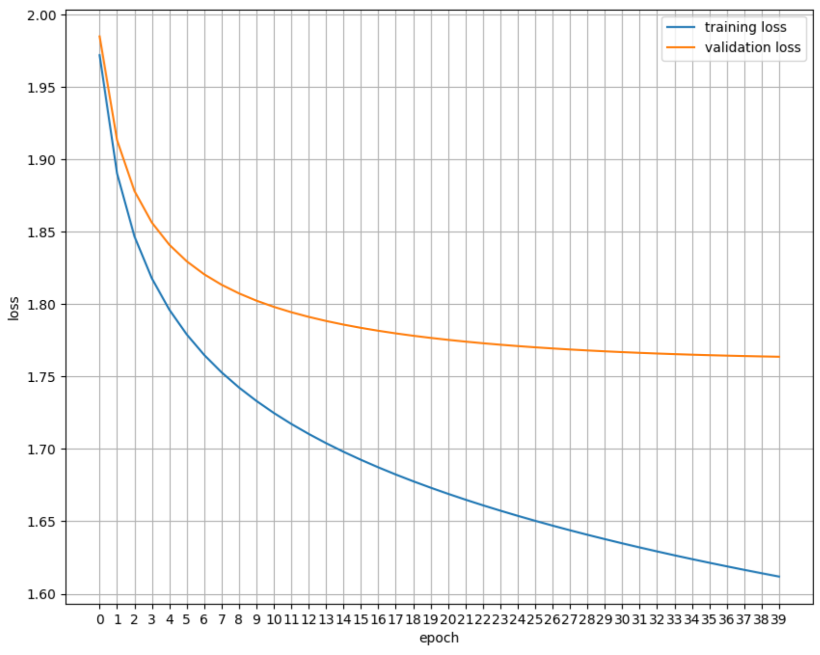

Figure 3: loss function for lambda=0, n_epochs=40, n_batch=100, eta=0.001

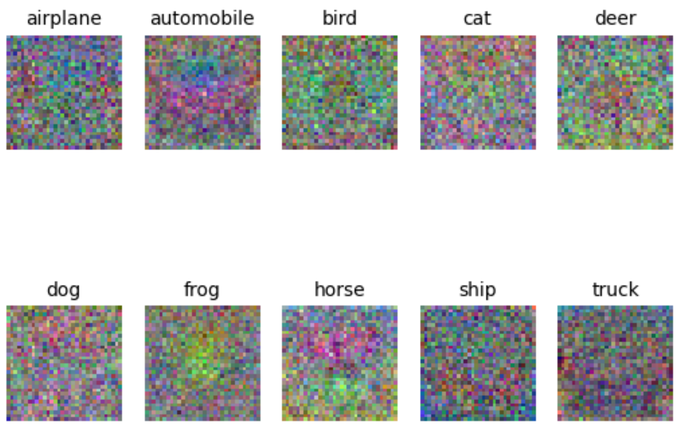

Figure 4: class templates for lambda=0, n_epochs=40, n_batch=100, eta=0.00

3. **lambda=0.1, n_epochs=40, n_batch=100, eta=0.001**

   which yielded the following accuracies:

- training set accuracy: 0.4464
- validation set accuracy: 0.3917
- test set accuracy: 0.3881

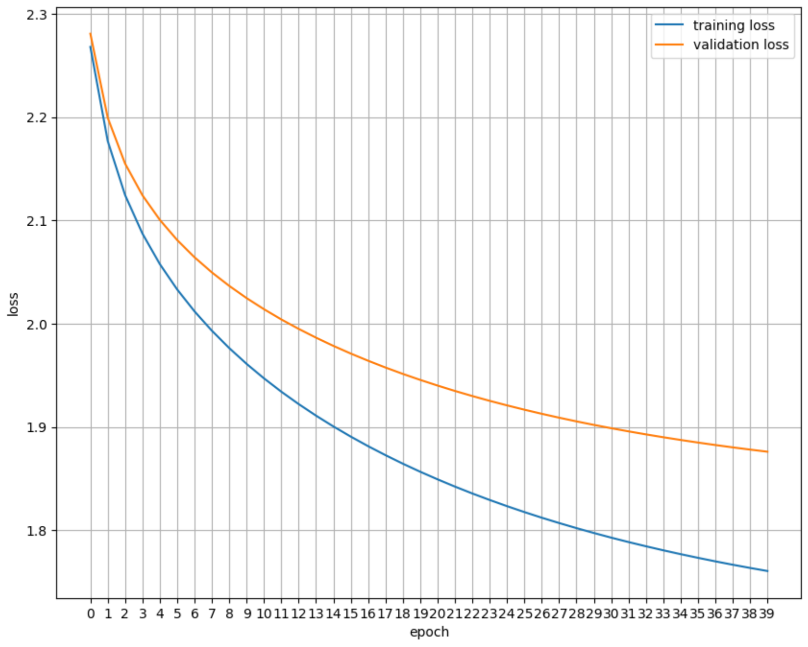

Figure 5: loss function for lambda=0.1, n_epochs=40, n_batch=100, eta=0.001

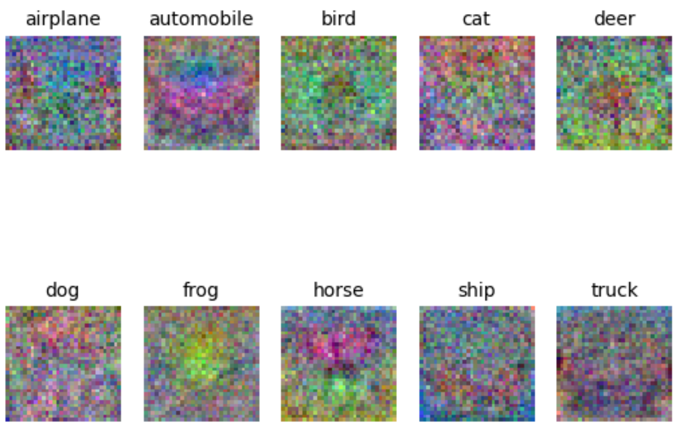

Figure 6: class templates for lambda=0.1, n_epochs=40, n_batch=100, eta=0.001

4. lambda=1, n_epochs=40, n_batch=100, eta=0.001

   which yielded the following accuracies:

- training set accuracy: 0.3985
- validation set accuracy: 0.3759
- test set accuracy: 0.3646

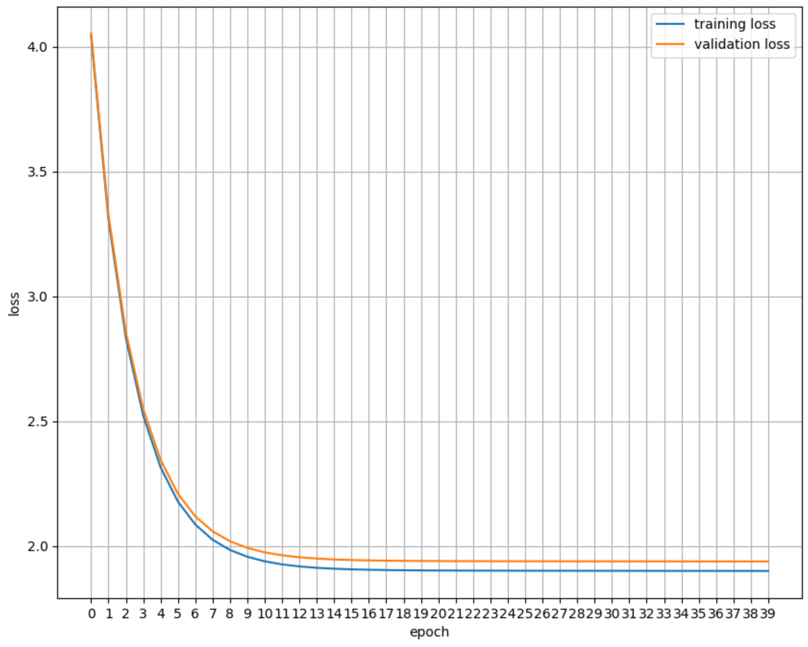

Figure 7: loss function for lambda=1, n_epochs=40, n_batch=100, eta=0.001

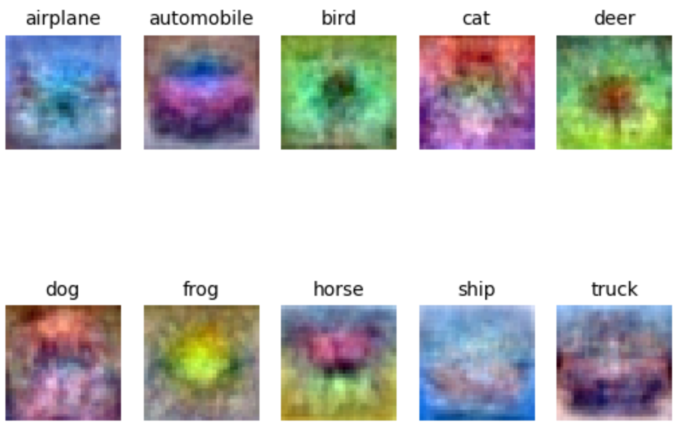

Figure 8: class templates for lambda=1, n_epochs=40, n_batch=100, eta=0.001

By inspecting the plots, one can conclude that a too high lambda results in the learning algorithm becoming unable to learn. This can be seen in Figure 4. One can also reach this conclusion by inspecting the training set accuracy of parameter set 4 and comparing it to the accuracies for the validation and test sets. Since the training set accuracy is roughly on par with the other two data set accuracies, one can conclude that the algorithm has not been able to learn properly.

Problems also arise when the learning rate is too large, as in Figure 1, where the results show a high degree of instability. In this case, the algorithm is unable to descend smoothly, due to it being forced to take too large leaps in its steps. For this parameter set, the class templates (displayed in Figure 2) display no patterns.

Finally, one sees that the higher one sets lambda (the regularization term) to be, the smoother the learned class templates become.