Clément Gousseau

DD 2424: Assignment 2

1. Checking the gradients

To check the gradients in a decent amount of time I reduced the dimensionality to d=200 and I used the first 10 inputs. I compared it with the results of ComputeGradientsSlow. Looking at the first coefficients it seemed to be right:

```
>> [grad_b{1}(1:10) grad_bnum{1}(1:10)]

ans =

   1.0e-03 *

    0.0400    0.0400
   -0.1936   -0.1936
    0.1074    0.1074
    0.1531    0.1531
    0.0480    0.0480
   -0.0791   -0.0791
   -0.0536   -0.0536
   -0.0161   -0.0161
    0.1791    0.1791
    0.0571    0.0571
```

```
>> [grad_b{2}(1:10) grad_bnum{2}(1:10)]

ans =

    0.0999    0.0999
   -0.0999   -0.0999
    0.0002    0.0002
   -0.0000   -0.0000
   -0.0000   -0.0000
    0.0999    0.0999
    0.0000    0.0000
   -0.0001   -0.0001
   -0.0000   -0.0000
   -0.0999   -0.0999
```

```
>> [grad_W{1}(1:10,1) grad_Wnum{1}(1:10,1)]

ans =

   -0.0012   -0.0012
    0.0023    0.0023
   -0.0015   -0.0015
   -0.0021   -0.0021
   -0.0018   -0.0018
   -0.0012   -0.0012
   -0.0011   -0.0011
    0.0003    0.0003
   -0.0004   -0.0004
    0.0011    0.0011
```

```
>> [grad_W{2}(1:10,1) grad_Wnum{2}(1:10,1)]

ans =

    0.0001    0.0001
    0.0004    0.0004
   -0.0004   -0.0004
   -0.0008   -0.0008
   -0.0009   -0.0009
    0.0005    0.0005
    0.0003    0.0003
    0.0022    0.0022
   -0.0038   -0.0038
   -0.0007   -0.0007
```

To be sure I computed the maximum relative error on all the coefficients:

```
>> max(abs((grad_b{1}-grad_bnum{1})./grad_bnum{1}))

ans =

   9.5551e-07

>> max(abs((grad_b{2}-grad_bnum{2})./grad_bnum{2}))

ans =

   1.1098e-04
```

```
>> max(max(abs((grad_W{1}-grad_Wnum{1})./grad_Wnum{1})))

ans =

   1.4439e-04

>> max(max(abs((grad_W{2}-grad_Wnum{2})./grad_Wnum{2})))

ans =

   2.5806e-05
```

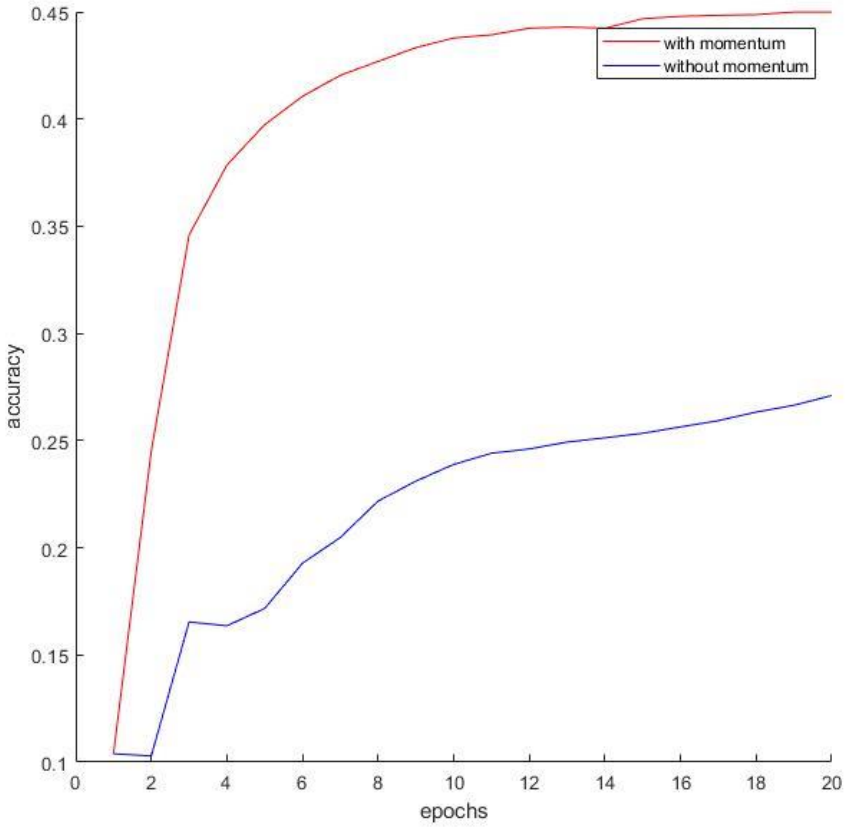
The error is always less than 0.001, which can be considered as acceptable.

2. Adding momentum

With eta=0.025, lambda=1e-5, n_batch=100.

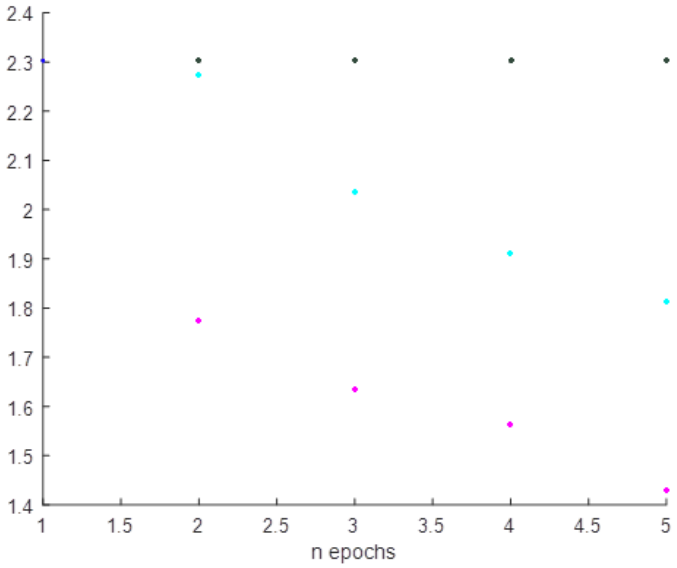
In blue, rho=0.9: the learning is pretty fast, after about 10 epochs the accuracy is about 44% and stable

In red, rho=0 (no momentum): the learning is very slow. Even after 20 epochs the accuracy is only about 25%



3. Finding the best eta/lambda

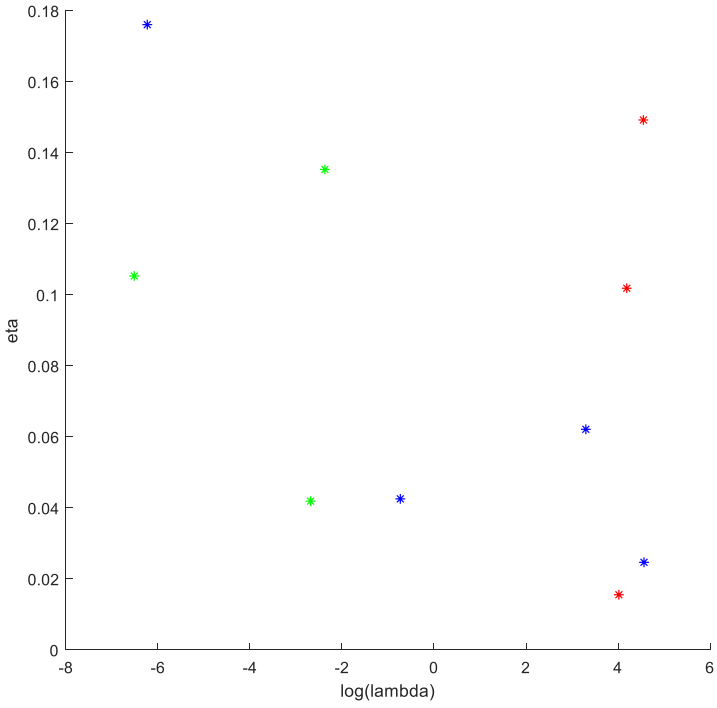
To find a good range for eta I tried to find a reasonable range for eta. I tried with $eta = 10^{-p}\ with\ -6 \leq p \leq 0$.



The black points correspond to eta=0.000001, eta=0.00001, eta=0.0001, eta=0.001 (the cost are so close that one cannot distinguish the points). The cyan points correspond to eta=0.01, the magenta points correspond to eta=0.1. For eta=1 the cost was increasing or Nan. The best eta seemed to be around 0.1.

Then I did a narrower search to find the best pair of eta/lambda

First search: $lambda \in [10^{-8}, 10^{-6}]$ and $eta \in [0.01,0.2]$ (10 pairs)

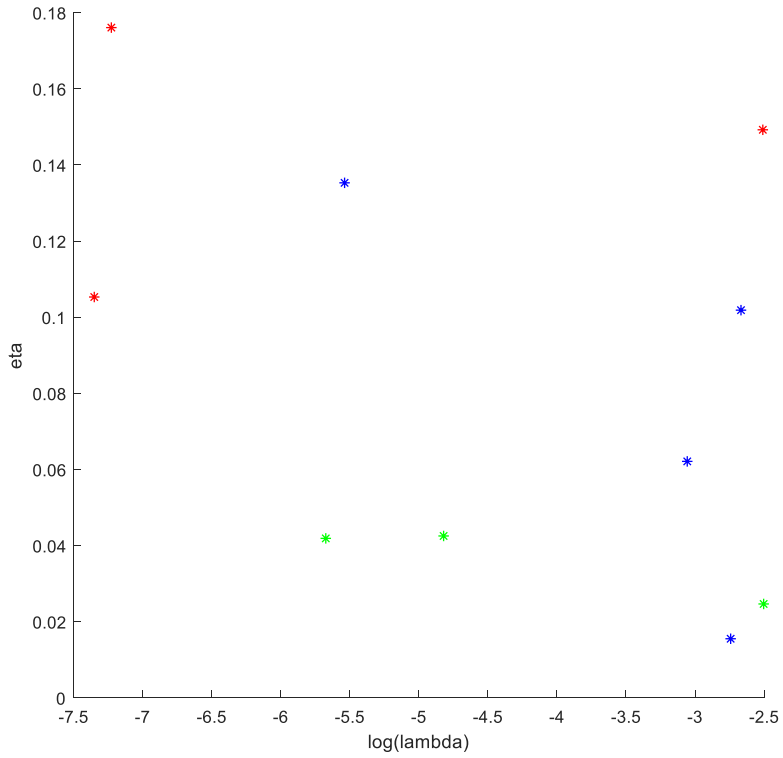


For each pair I computed the accuracy on the test set and sorted the results

The 25% highest accuracies are in green
The 25% lowest accuracies are in red
The others are in blue

The accuracy seems to be lower with small lambdas.

Second search: $lambda \in [10^{-8}, 10^{-2}]$ and $eta \in [0.01,0.2]$ (10 pairs)

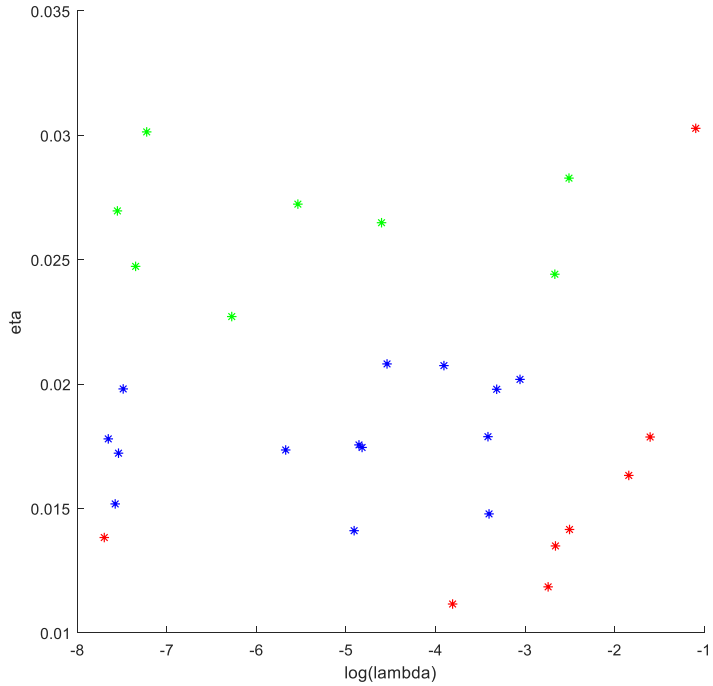


For each pair I computed the accuracy on the test set and sorted the results

The 25% highest accuracies are in green
The 25% lowest accuracies are in red
The others are in blue

The accuracy seems to be lower with small eta. With this range of lambda it does not seem there is a high dependency on lambda

Third search: $lambda \in [10^{-8}, 10^{-2}]$ and $eta \in [0.01,0.3]$ (30 pairs)



For each pair I computed the accuracy on the test set and sorted the results

The 25% highest accuracies are in green
The 25% lowest accuracies are in red
The others are in blue

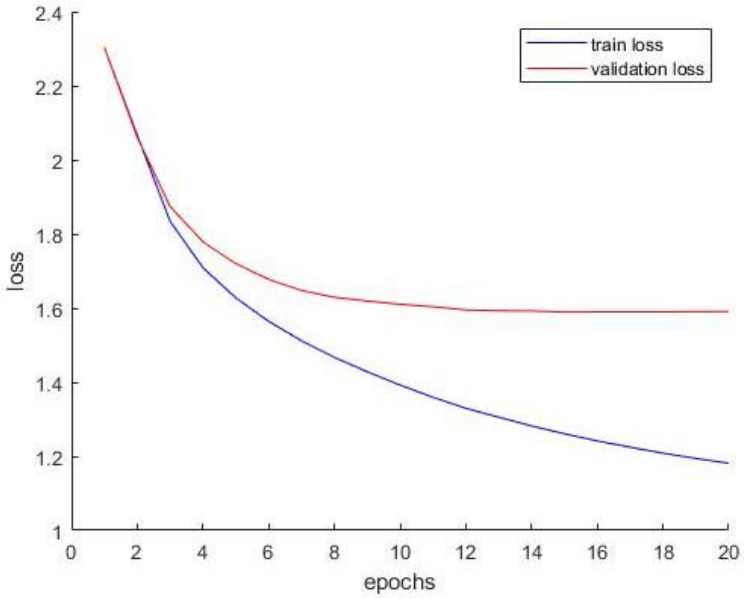
The accuracy seems to be lower with eta about 0.025. With this range of eta it does not seem there is a high dependency on lambda.

For the rest of the assignment I chose eta=0.025, lambda=1e-5.

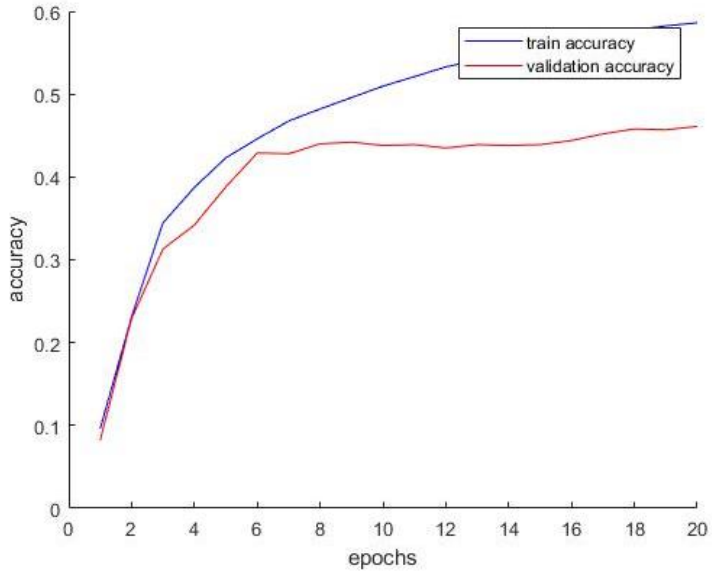
4. Results with the best parameter setting

The final parameter are eta=0.025, lambda=1e-5, nbatch=100, rho=0.9, decay rate=0.9

I trained the network on all training data except a validation set of 1000 data.



As expected the loss decreases faster on the training set than on the validation set. After about 10 epochs the loss is stable on the validation set.



The final accuracy is 58.6% on the training set, 46.1% on the validation set

Performance of the best network on the test set: the accuracy is 44.6%