Assignment 2: Report basic

1) Checking the gradients

In order to check the correctness of the gradients I used the following formula

$$\frac{|g_a - g_n|}{\max(\text{eps}, |g_a| + |g_n|)} \quad \text{where eps a very small positive number}$$

where g_a is the analytically computed gradient and g_n is the numerically computed gradient and made sure that this relative error is small for each component of the gradient(bias gradient and weights gradient). To compute the numerical gradients I used the function ***ComputeGradsNumSlow.py.***

First Layer

Weigh matrix

```
- sum of abs differences: 7.833e-03
- mean of abs values   grad: 1.209e-01   grad_num: 1.209e-01
- min  of abs values   grad: 1.457e-06   grad_num: 1.251e-06
- max  of abs values   grad: 6.873e-01   grad_num: 6.873e-01
```

Bias matrix

```
- sum of abs differences: 5.684e-07
- mean of abs values   grad: 1.316e-01   grad_num: 1.316e-01
- min  of abs values   grad: 4.711e-02   grad_num: 4.711e-02
- max  of abs values   grad: 4.574e-01   grad_num: 4.574e-01
```

SecondLayer

Weigh matrix 2

```
- sum of abs differences: 7.442e-05
- mean of abs values   grad: 5.806e-01   grad_num: 5.806e-01
- min  of abs values   grad: 5.705e-04   grad_num: 5.702e-04
- max  of abs values   grad: 8.951e+00   grad_num: 8.951e+00
```

Bias matrix 2

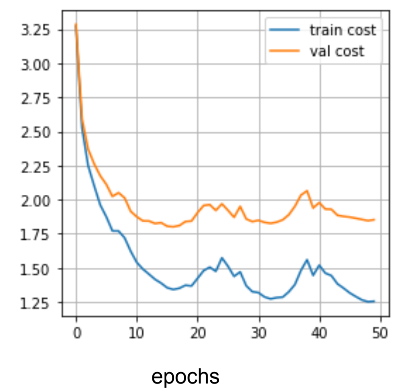
```
- sum of abs differences: 2.324e-07
- mean of abs values   grad: 1.697e-01   grad_num: 1.697e-01
- min  of abs values   grad: 1.754e-02   grad_num: 1.754e-02
- max  of abs values   grad: 8.484e-01   grad_num: 8.484e-01
```

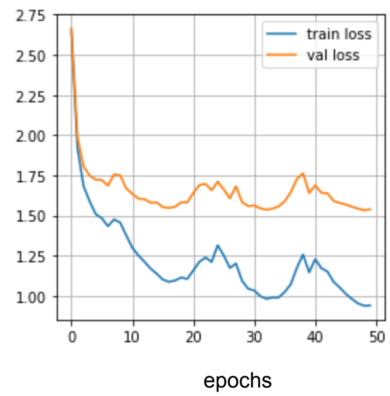
From the slow relative error we can conclude that the gradients are implemented correctly.

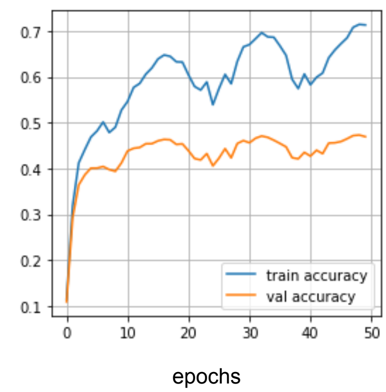
2) Training and validation loss/cost when using cyclical learning rates

In the following the curves for the validation, loss and accuracy of the training and validation set are shown. The network is a 2-layer NN with 50 hidden nodes. The training is done for 50 epochs with batch size 100, lambda=0 and eta_min = 1e-5, eta_max=1e-1 and n_s = 800.

The cyclical learning rate is represented on the rhythmic cycles of the curves every 2n_s.







The final results are:

Train set accuracy: 0.6989

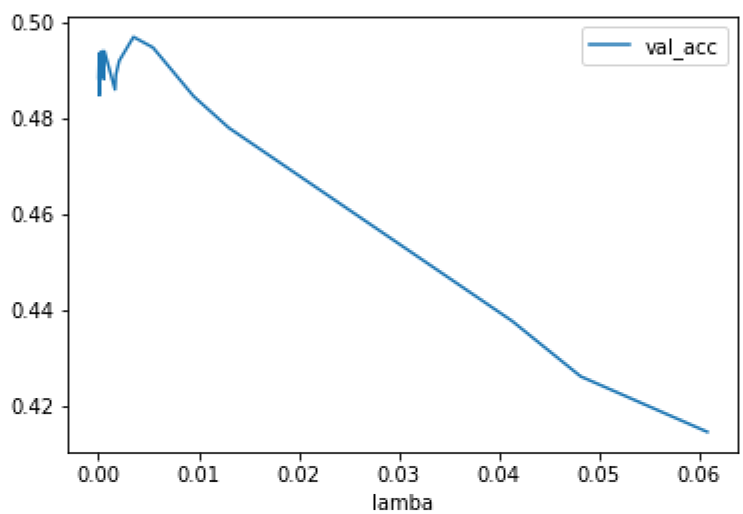
Validation set accuracy: 0.4638

Test set accuracy: 0.4619

3) Coarse search for lambda

In order to choose the right hyperparameter for lambda for the neural network mentioned in part 2 we do a coarse to fine search by sampling a lambda between 10^l_min and 10^l_max. Since we have to properly train the network later we search for the best lambda while using **the full data set** leaving out only 5000 samples for the validation set. For each lambda the neuronal network is trained for 4 epochs. Since n_s=900 and batch size is 100 with a dataset of 45000 samples, this corresponds to **2 cycles**.

Coarse search: l_min = -5 and l_max=-1.

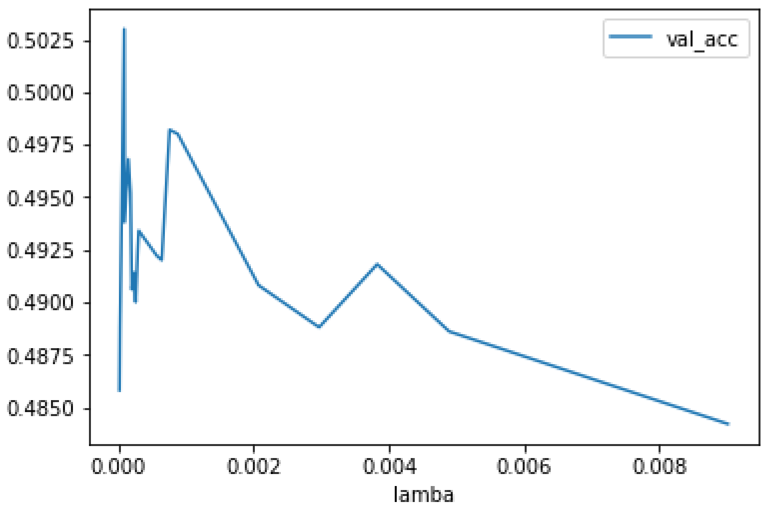


lamba	val_acc	train_acc
0.060834	0.4144	0.428067
0.048186	0.4260	0.441400
0.041328	0.4376	0.456000
0.013022	0.4780	0.516267
0.009508	0.4846	0.527200
0.000087	0.4848	0.553933
0.001653	0.4860	0.549356
0.000531	0.4882	0.554978
0.000013	0.4884	0.554133
0.000087	0.4886	0.552978
0.001734	0.4890	0.552422
0.000061	0.4912	0.554467
0.000054	0.4916	0.556289
0.002071	0.4920	0.549356
0.000064	0.4936	0.553867
0.000273	0.4938	0.551311
0.000398	0.4940	0.552933
0.000596	0.4940	0.554667
0.005441	0.4948	0.542422
0.003500	0.4970	0.547311

The three best networks deliver validation accuracies of 0.4970, 0.4948 and 0.4940 respectively.

4) Fine search for lambda

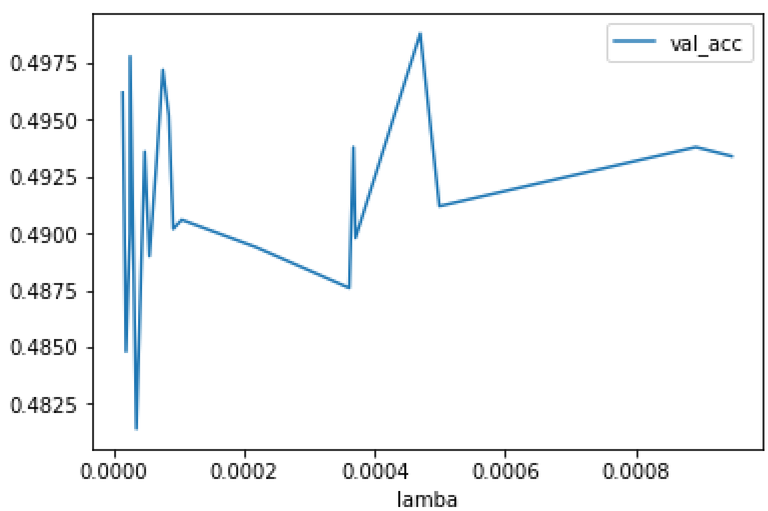
Fine search 1: l_min = -5 and l_max=-2.



lamba	val_acc	train_acc
0.009015	0.4842	0.527267
0.000012	0.4858	0.553978
0.004893	0.4886	0.538844
0.002967	0.4888	0.543444
0.000020	0.4888	0.551711
0.000251	0.4900	0.555889
0.000194	0.4906	0.553156
0.002071	0.4908	0.552378
0.000228	0.4914	0.556733
0.003826	0.4918	0.541644
0.000637	0.4920	0.552756
0.000566	0.4922	0.554733
0.000281	0.4924	0.553867
0.000297	0.4934	0.555689
0.000088	0.4938	0.555689
0.000175	0.4952	0.553689
0.000143	0.4968	0.554156
0.000875	0.4980	0.554911
0.000754	0.4982	0.558667

The three best networks deliver validation accuracies of 0.4982, 0.4980 and 0.4968 respectively.

Fine search l_min = -5 and l_max=-3



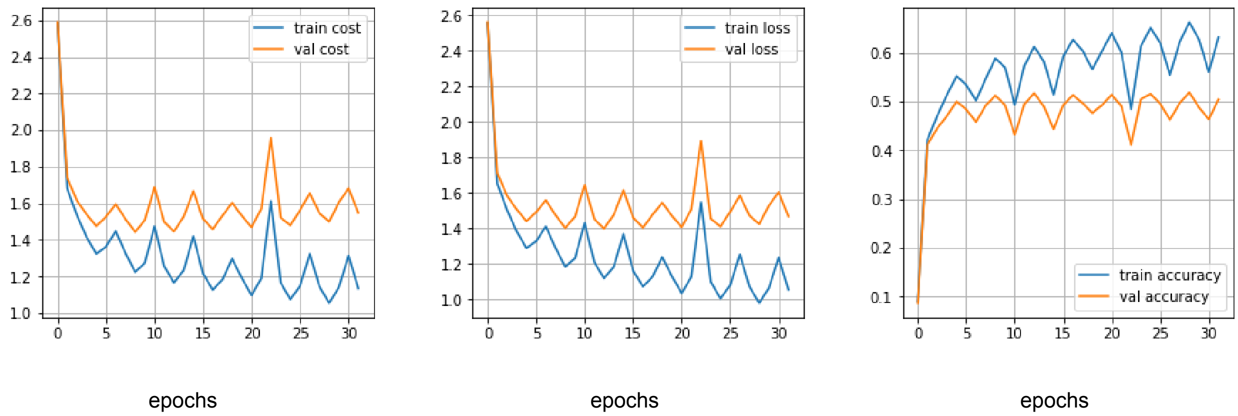
lamba	val_acc	train_acc
0.000035	0.4814	0.555467
0.000019	0.4848	0.553244
0.000361	0.4876	0.557622
0.000055	0.4890	0.556022
0.000218	0.4894	0.556089
0.000371	0.4898	0.553400
0.000025	0.4900	0.558067
0.000091	0.4902	0.552978
0.000104	0.4906	0.553978
0.000499	0.4912	0.552556
0.000066	0.4932	0.555044
0.000948	0.4934	0.556644
0.000048	0.4936	0.553689
0.000893	0.4938	0.552267
0.000368	0.4938	0.555489
0.000085	0.4952	0.556644
0.000014	0.4962	0.556644
0.000075	0.4972	0.558511
0.000025	0.4978	0.556067
0.000470	0.4988	0.554933

The three best networks deliver validation accuracies of 0.4988, 0.4978 and 0.4972 respectively.

In conclusion the best accuracy in the validation set is achieved by lambda=0.00047.

5) Full training set with previously found lambda

The 2-layer network with 50 hidden nodes is trained for 32 epochs which corresponds to 8 cycles using the following parameters:
lambda = 0.000470, eta_min=1e-5, eta_max=1e-1, n_s=900



It delivers the following accuracies:
Train set accuracy: 0.6692222222222223
Validation set accuracy: 0.5188
Test set accuracy: 0.5117