MSc Machine Learning

DD2424 Deep Learning in Data Science

Assignment 3: Image classification with a k-layer network

Author:
Alexandros Ferles
ferles@kth.se

Professor:
Josephine Sullivan

Contents

1 Compute the gradients of the network parameters **2**

2 Can I train a 3-layer network? **3**

3 Implement Batch Normalization **8**

1 Compute the gradients of the network parameters

Upgrade Assignment 2 code to train test k-layer net-works

In order to confirm that all previous code was extended succesfully to k-layers, we test that for 2,3 and 4 layers respectively we can compute gradients that differ by a small amount compared to their numerical ones:

1. For a 2-layer network we get:

```
-----------------
Layer no. 1:
Deviation on weight matrix: 7.52283628908819e-09
Deviation on bias vector: 4.602426291547493e-09
-----------------
Layer no. 2:
Deviation on weight matrix: 3.0747216543846533e-10
Deviation on bias vector: 1.1936382115255306e-11
```

2. For a 3-layer network we get:

```
-----------------
Layer no. 1:
Deviation on weight matrix: 2.1237177447134792e-06
Deviation on bias vector: 1.047827661340416e-06
-----------------
Layer no. 2:
Deviation on weight matrix: 1.197615459525251e-07
Deviation on bias vector: 0.0032312506974258464
-----------------
Layer no. 3:
Deviation on weight matrix: 7.368229647407527e-08
Deviation on bias vector: 1.677023298915395e-11
```

3. For a 4-layer network we get:

```
-----------------
Layer no. 1:
Deviation on weight matrix: 0.0004455622900617491
Deviation on bias vector: 0.00021413577255454145
-----------------
Layer no. 2:
Deviation on weight matrix: 2.8193570737353638e-05
Deviation on bias vector: 0.014347056990204365
-----------------
Layer no. 3:
Deviation on weight matrix: 8.838559097221378e-06
Deviation on bias vector: 0.17027488699016166
-----------------
Layer no. 4:
Deviation on weight matrix: 2.0539890893159423e-05
Deviation on bias vector: 9.891207642802771e-12
```

2 Can I train a 3-layer network?

First check, with the new version of your code, you can replicate the (default) results you achieved in Assignment 2 with a 2-layer network with 50 nodes in the hidden layer.

For a 2-layer network, and by using the same setting we get a test-set accuracy performance of 44.16%, which due to the randomness on the initialization on the weights shows us that the extension was succesfully implemented (in the pprevious assignment by using the same settings the highest test set accuracy achieved was 44.4%).

Your next task is to try and train a 3-layer network with 50 and 30 nodes in the first and second hidden layer respectively with the same learning parameters. What happens after a few epochs of training? Are you learning anything? What happens if you play around with the learning rate eta? What happens if you use He initialization?

Firstly, we show the training set cross-entropy loss evolution for a 3-layer network:

```
Cost at training epoch 1 is 2.3022672535670137
Cost at training epoch 2 is 2.3022666352616192
Cost at training epoch 3 is 2.302262164108896
Cost at training epoch 4 is 2.302256330131588
Cost at training epoch 5 is 2.3022509063293954
Cost at training epoch 6 is 2.302246087045059
Cost at training epoch 7 is 2.30224183447161
Cost at training epoch 8 is 2.3022380835454994
Cost at training epoch 9 is 2.3022347717577407
Cost at training epoch 10 is 2.3022318440902114
Cost at training epoch 11 is 2.3022292524125856
Cost at training epoch 12 is 2.3022269539491447
Cost at training epoch 13 is 2.3022249115918183
Cost at training epoch 14 is 2.302223092730126
Cost at training epoch 15 is 2.3022214685589133
```

Since the training loss barely cahnges after 10 epochs of training, we try different values for the learning rate η:

```
------------------
Eta:  1e-06
100%|██████████| 10/10 [00:18<00:00,  1.89s/it]
Cost at training epoch 1 is 2.30260046125
Cost at training epoch 2 is 2.3026003916138555
Cost at training epoch 3 is 2.302600325471149
Cost at training epoch 4 is 2.3026002626463438
Cost at training epoch 5 is 2.3026002029724792
Cost at training epoch 6 is 2.3026001462910486
Cost at training epoch 7 is 2.3026000924515975
Cost at training epoch 8 is 2.302600041311279
Cost at training epoch 9 is 2.3025999927344216
Cost at training epoch 10 is 2.3025999465922684
------------------
Eta:  1e-05
100%|██████████| 10/10 [00:17<00:00,  1.73s/it]
Cost at training epoch 1 is 2.3025999005718703
Cost at training epoch 2 is 2.302599205749114
Cost at training epoch 3 is 2.3025985468529453
Cost at training epoch 4 is 2.3025979219724637
Cost at training epoch 5 is 2.302597329300965
Cost at training epoch 6 is 2.302596767131986
Cost at training epoch 7 is 2.3025962338540733
Cost at training epoch 8 is 2.302595727947948
Cost at training epoch 9 is 2.3025952479739393
Cost at training epoch 10 is 2.3025947925707957
------------------
Eta:  0.0001
100%|██████████| 10/10 [00:17<00:00,  1.72s/it]
Cost at training epoch 1 is 2.3025943342251414
Cost at training epoch 2 is 2.30258753778816
Cost at training epoch 3 is 2.3025811965366985
Cost at training epoch 4 is 2.302575274681294
Cost at training epoch 5 is 2.3025697397215943
Cost at training epoch 6 is 2.3025645621776087
Cost at training epoch 7 is 2.3025597151526602
Cost at training epoch 8 is 2.302555174170237
Cost at training epoch 9 is 2.302550916949251
Cost at training epoch 10 is 2.302546923101441
------------------
Eta:  0.001
100%|██████████| 10/10 [00:16<00:00,  1.70s/it]
Cost at training epoch 1 is 2.302542530097267
Cost at training epoch 2 is 2.3024880377323287
Cost at training epoch 3 is 2.302444833211863
Cost at training epoch 4 is 2.3024102404890057
Cost at training epoch 5 is 2.302382282712888
Cost at training epoch 6 is 2.3023594877124416
Cost at training epoch 7 is 2.3023407475778948
Cost at training epoch 8 is 2.302325219905004
Cost at training epoch 9 is 2.3023122586043185
Cost at training epoch 10 is 2.3023013637986485
```

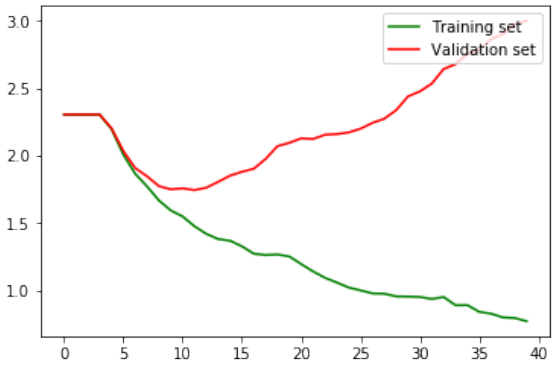
```
Eta:  0.001
100%|██████████| 10/10 [00:16<00:00,  1.70s/it]
Cost at training epoch 1 is 2.302542530097267
Cost at training epoch 2 is 2.3024880377323287
Cost at training epoch 3 is 2.302444833211863
Cost at training epoch 4 is 2.3024102404890057
Cost at training epoch 5 is 2.302382282712888
Cost at training epoch 6 is 2.3023594877124416
Cost at training epoch 7 is 2.3023407475778948
Cost at training epoch 8 is 2.302325219905004
Cost at training epoch 9 is 2.3023122586043185
Cost at training epoch 10 is 2.3023013637986485
------------------
Eta:  0.01
100%|██████████| 10/10 [00:17<00:00,  1.73s/it]
Cost at training epoch 1 is 2.3022832235898933
Cost at training epoch 2 is 2.3022314934409445
Cost at training epoch 3 is 2.3022246500421755
Cost at training epoch 4 is 2.302222636213854
Cost at training epoch 5 is 2.302221210438927
Cost at training epoch 6 is 2.302219944464948
Cost at training epoch 7 is 2.3022188047640544
Cost at training epoch 8 is 2.3022177857331054
Cost at training epoch 9 is 2.3022168786733173
Cost at training epoch 10 is 2.3022160723738483
------------------
Eta:  0.1
100%|██████████| 10/10 [00:18<00:00,  1.90s/it]
Cost at training epoch 1 is 2.304083147255321
Cost at training epoch 2 is 2.3040527762143457
Cost at training epoch 3 is 2.3040247984649436
Cost at training epoch 4 is 2.303963513926076
Cost at training epoch 5 is 2.1974427096362485
Cost at training epoch 6 is 2.007216352396664
Cost at training epoch 7 is 1.8656019902160936
Cost at training epoch 8 is 1.7714920839464587
Cost at training epoch 9 is 1.6672864486126826
Cost at training epoch 10 is 1.593195430942723
```

Again, the loss remains at the same levels no matter how many update steps take part in the training! Only with a fairly big learning rate of 0.1 we can see a small drop in the loss, but again a few epochs in the beginning where the loss remain stable can be observed! We can also suspect that with such a big learning rate, the learning might be unstable.

The last experiment is conducted for more epochs:

```
Cost at training epoch 1 is 2.304083147255321
Cost at training epoch 2 is 2.3040527762143457
Cost at training epoch 3 is 2.3040247984649436
Cost at training epoch 4 is 2.303963513926076
Cost at training epoch 5 is 2.1974427096362485
Cost at training epoch 6 is 2.007216352396664
Cost at training epoch 7 is 1.8656019902160936
Cost at training epoch 8 is 1.7714920839464587
Cost at training epoch 9 is 1.6672864486126826
Cost at training epoch 10 is 1.593195430942723
Cost at training epoch 11 is 1.5485123476073677
Cost at training epoch 12 is 1.4758706564115107
Cost at training epoch 13 is 1.4193924744826478
Cost at training epoch 14 is 1.3805438066881317
Cost at training epoch 15 is 1.368697249854167
Cost at training epoch 16 is 1.3265087669239681
Cost at training epoch 17 is 1.2723064324945854
Cost at training epoch 18 is 1.2625014563672678
Cost at training epoch 19 is 1.2666867686005
Cost at training epoch 20 is 1.25163518706638
Cost at training epoch 21 is 1.1950692780730536
Cost at training epoch 22 is 1.1405550951346806
Cost at training epoch 23 is 1.0933375948769273
Cost at training epoch 24 is 1.0584354555304865
Cost at training epoch 25 is 1.0211788369490433
Cost at training epoch 26 is 1.0002588070688778
Cost at training epoch 27 is 0.9777619847054716
Cost at training epoch 28 is 0.9753313784903359
Cost at training epoch 29 is 0.9558171620564835
Cost at training epoch 30 is 0.9544156273418551
Cost at training epoch 31 is 0.9515871093045444
Cost at training epoch 32 is 0.93641149448918
Cost at training epoch 33 is 0.9511512546042148
Cost at training epoch 34 is 0.8900400762553463
Cost at training epoch 35 is 0.891207534551131
Cost at training epoch 36 is 0.840768908056229
Cost at training epoch 37 is 0.8274775595253576
Cost at training epoch 38 is 0.799688813021933
Cost at training epoch 39 is 0.7950914142037012
Cost at training epoch 40 is 0.7716458010015995
```

With a cross-entropy loss evolution:



We observe overfitting with this setting.

Last but not least, we switch to He initialization:

```
Cost of training with He initialization at epoch 1 is 2.573371144365728
Cost of training with He initialization at epoch 2 is 2.4388440346442506
Cost of training with He initialization at epoch 3 is 2.3685662732480903
Cost of training with He initialization at epoch 4 is 2.3142257340862384
Cost of training with He initialization at epoch 5 is 2.272922477058916
Cost of training with He initialization at epoch 6 is 2.236912218216597
Cost of training with He initialization at epoch 7 is 2.2052890641621268
Cost of training with He initialization at epoch 8 is 2.1762496203378463
Cost of training with He initialization at epoch 9 is 2.1514607505444694
Cost of training with He initialization at epoch 10 is 2.1297865298452456
```

The loss seems to be dropping instead of staying stable, but still the drop observed is very slow:

3 Implement Batch Normalization

Check your analytical gradients computation in comparison with the numerical ones for 2 layers and 50 hidden nodes

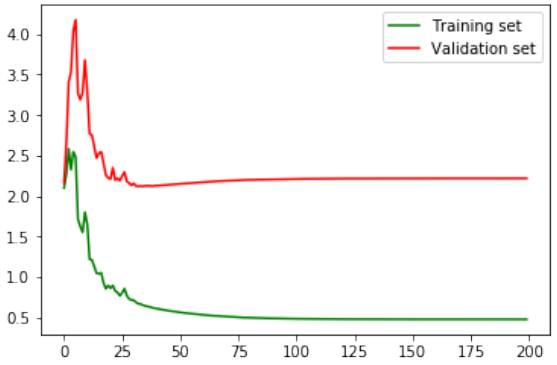
- For 2 layers:

```
-----------------
Layer no. 1:
Deviation on weight matrix: 1.2597928505427309e-08
Deviation on bias vector: 0.04489583333333333
-----------------
Layer no. 2:
Deviation on weight matrix: 1.061126355032502e-09
Deviation on bias vector: 1.085592909420537e-09
```

- For 3 layers:

```
-----------------
Layer no. 1:
Deviation on weight matrix: 1.4574774508711658e-05
Deviation on bias vector: 0.020078843569924645
-----------------
Layer no. 2:
Deviation on weight matrix: 0.0006714264687067433
Deviation on bias vector: 0.1015625
-----------------
Layer no. 3:
Deviation on weight matrix: 1.0708749294639795e-11
Deviation on bias vector: 1.2260625257832792e-11
```

We also conduct a sanity check, trying to overfit a 3-layer network in a few training data:



Overfitting is achieved, since the accuracy in the training set is 87.7% and in the validation set it is 29.1%.

Include graphs of the evolution of the loss function when you tried to train your 3-layer network without batch normalization and with batch normalization.

When trying to train the network without applying batch normalization, we observe the following cross-entropy loss and accuracy evolution to the training and validation set:

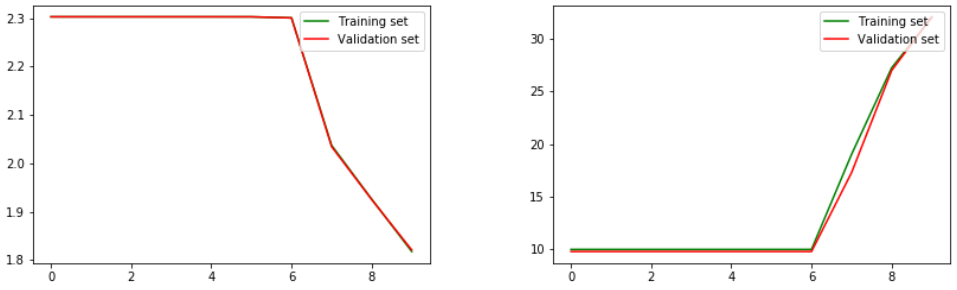


Figure 1: Cross-entropy loss evolution (left) and accuracy performance evolution (right) when batch normalization is **not** applied.

The accuracy achieved in the test set data is 25.63%.

We conduct training with the same hyperparameters $(\eta, \lambda) = (0.012913581489067944, 10e^{-4})$ and this time batch normalization is applied:

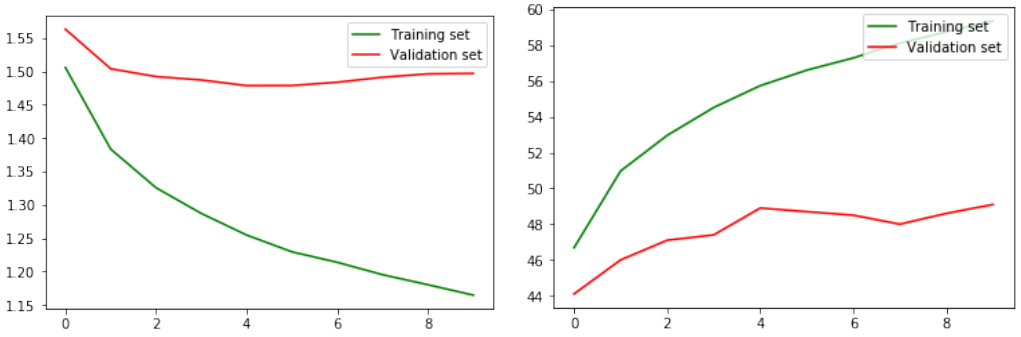


Figure 2: Cross-entropy loss evolution (left) and accuracy performance evolution (right) when batch normalization **is** applied.

This time an accuracy performance of 50.84% is reported in the test set.

State the range of the values you searched for lambda and eta, the number of epochs used for training during the fine search, and the hyper-parameter settings for your best performing 3-layer network you trained with batch normalization. Also state the test accuracy achieved by network.

Coarse search:

For λ between 10^{-6} and 10^{-2} the search for appropriate η was performed in [0.005,0.1]

Fine search:

For $\lambda = 10^{-6}$ the search for appropriate η was performed in [0.0057, 0.019]

For $\lambda = 10^{-5}$ the search for appropriate η was performed in [0.007, 0.015]

For $\lambda = 10^{-4}$ the search for appropriate η was performed in [0.03, 0.06] $\bigcup$ [0.01, 0.015]

For $\lambda = 10^{-3}$ the search for appropriate η was performed in [0.008, 0.12]

For $\lambda = 10^{-2}$ the search for appropriate η was performed in [0.005, 0.007]

Best setting was $(\eta, \lambda) = (0.012913581489067944, 10^{-4})$ with a test set accuracy performance of 50.84

We also persent the cross-entropy loss evolution and the accuracy performance evolution for 2 other good pairs of (η, λ):

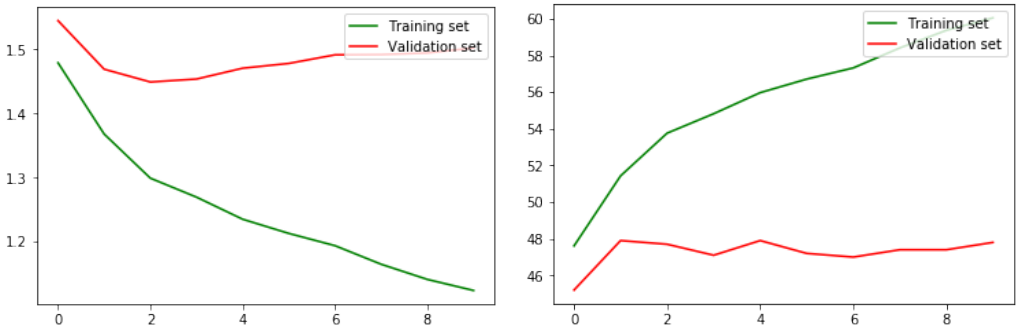


Figure 3: Cross-entropy loss evolution (left) and accuracy performance evolution (right) at a 3-layer with batch normalization and $\eta = 0.034875895633392565, \lambda = 10^{-5}$

Test set accuracy performance: 48.83%

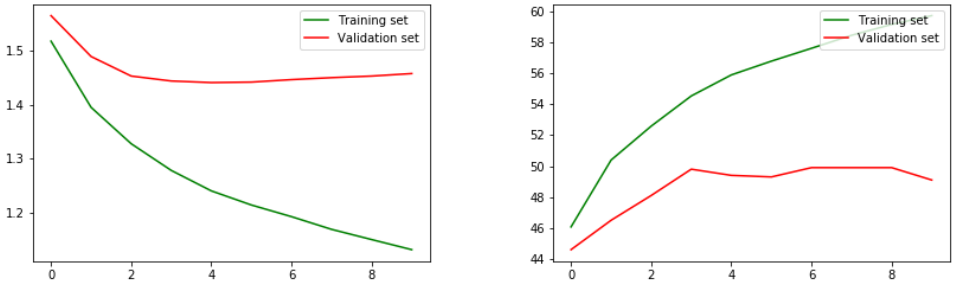


Figure 4: Cross-entropy loss evolution (left) and accuracy performance evolution (right) at a 2-layer with batch normalization and $\eta = 0.007986719995840757, \lambda = 10^{-6}$

Test set accuracy performance: 50.71%

Plot the training and validation loss for your 2-layer network with batch normalization with 3 different learning rates (small, medium, high) for 10 epochs and make the same plots for a 2-layer network with no batch normalization.

- Small learning rate $\eta = 0.0018920249916784752$:

  Without batch normalization:

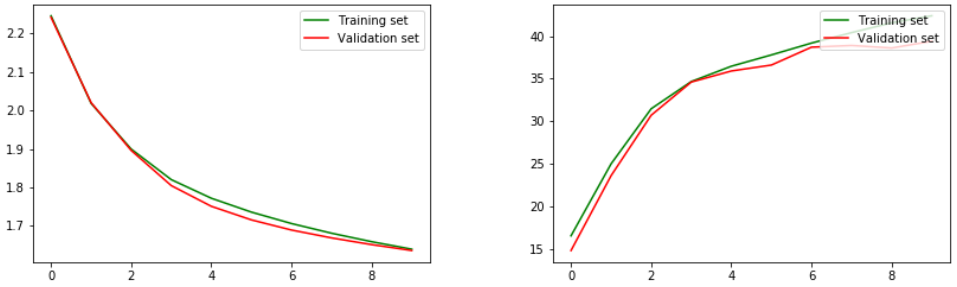


Figure 5: Cross-entropy loss evolution (left) and accuracy performance evolution (right) at a 2-layer **without** batch normalization and $\eta = 0.0018920249916784752$

Test set accuracy performance: 42.17%

With batch normalization:

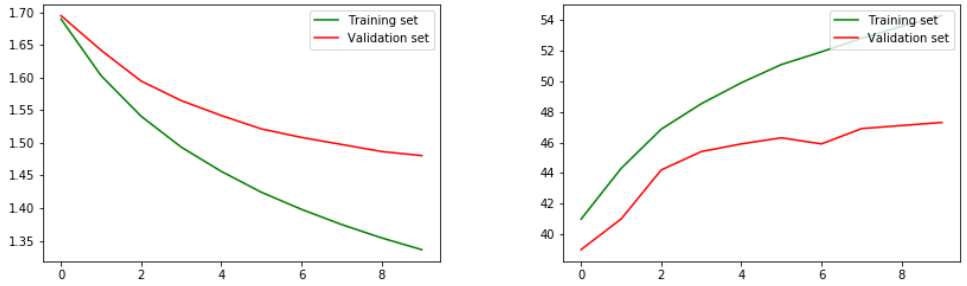


Figure 6: Cross-entropy loss evolution (left) and accuracy performance evolution (right) at a 2-layer **with** batch normalization and $\eta = 0.0018920249916784752$

Test set accuracy performance: 48.25%

- Medium learning rate $\eta = 0.1$:

Without batch normalization:

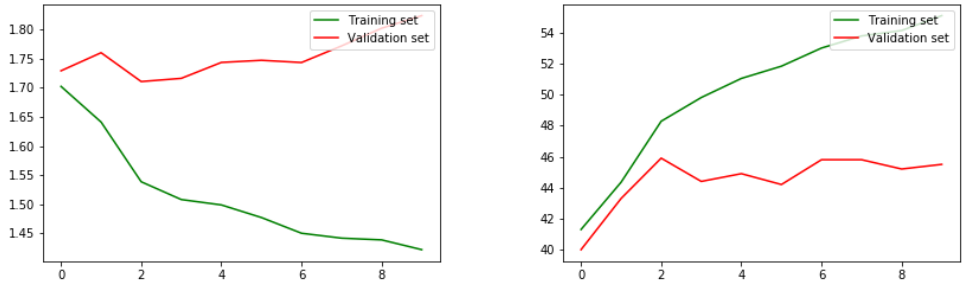


Figure 7: Cross-entropy loss evolution (left) and accuracy performance evolution (right) at a 2-layer **without** batch normalization and $\eta = 0.1$

Test set accuracy performance: 45%

With batch normalization:

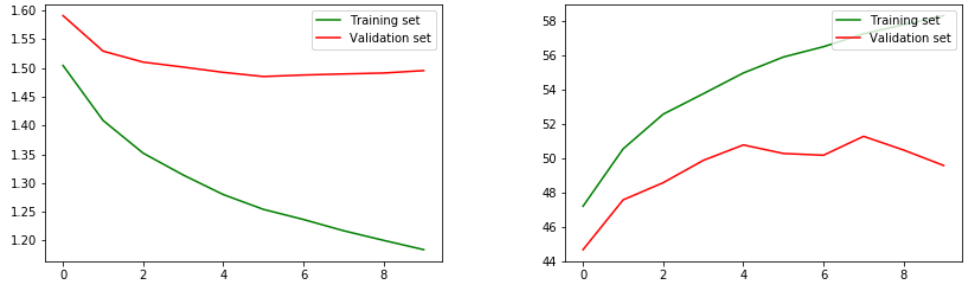


Figure 8: Cross-entropy loss evolution (left) and accuracy performance evolution (right) at a 2-layer **with** batch normalization and $\eta = 0.1$

Test set accuracy performance: 49.45%

- Medium learning rate $\eta = 0.1$:

Without batch normalization:

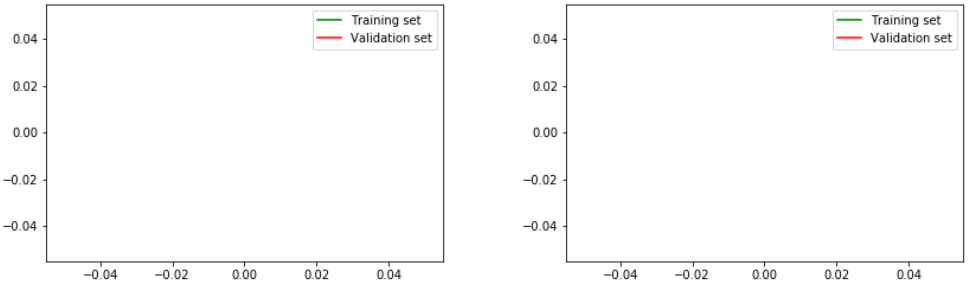


Figure 9: Cross-entropy loss evolution (left) and accuracy performance evolution (right) at a 2-layer **without** batch normalization and $\eta = 0.6$

Overflow error!
Test set accuracy performance: 18.22%

With batch normalization:

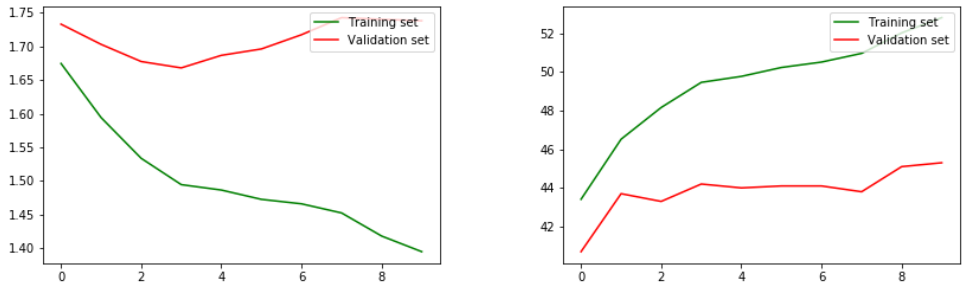


Figure 10: Cross-entropy loss evolution (left) and accuracy performance evolution (right) at a 2-layer **with** batch normalization and $\eta = 0.6$

Test set accuracy performance: 45.17%