DD2424 Deep Learning in Data Science

Name: Sri Janani Rengarajan *Programme: Embedded Systems*

Assignment 3: Training and testing of a K layer network with Batch Normalisation

The aim of the assignment is to classify images into ten classes using a neural network with multiple hidden layers. Input to the classifier is the vector **x,** which contains the information about the image. Output of the classifier is a probability vector **p**, for each possible class. The parameters of the classifier are **W, b,** $\boldsymbol{\gamma}$, $\boldsymbol{\beta}$**.** These parameters are learned by minimizing the cross-entropy loss of the classifier.

K- layer network implementation:

1) **Gradient computation check:**

   Analytical gradient computation was compared against numerical gradient computation for different batch sizes and ε values. The error results are presented in tables 1, 2 and 3. The error values are considerably small. Hence, we can conclude that the analytical method works well.

ε	$\frac{\partial J}{\partial W_1}$	$\frac{\partial J}{\partial W_2}$	$\frac{\partial J}{\partial b_1}$	$\frac{\partial J}{\partial b_2}$
0.00001	3.8360e-06	1.0666e-08	2.4260e-10	9.7576e-11
0.0001	3.8360e-07	1.0666e-08	2.4260e-10	9.7576e-11
0.001	3.8360e-08	1.0666e-08	2.4260e-10	9.7576e-11
0.01	3.8360e-09	1.5142e-09	2.4260e-10	9.7576e-11

Table 1. 2 layer network: Gradient computation error between analytical and numerical methods for batch size 100

ε	0.00001	0.0001	0.001	0.01
$\frac{\partial J}{\partial W_1}$	7.1791e-06	7.1791e-07	7.1791e-08	7.1791e-09
$\frac{\partial J}{\partial W_2}$	4.3858e-06	4.3858e-07	4.3858e-08	4.3858e-09
$\frac{\partial J}{\partial W_3}$	9.9249e-09	9.9249e-09	9.9249e-09	2.5905e-09
$\frac{\partial J}{\partial b_1}$	1.0187e-09	1.0187e-09	1.0187e-09	1.0187e-09
$\frac{\partial J}{\partial b_2}$	2.0072e-08	2.0072e-08	2.0072e-08	3.1208e-09
$\frac{\partial J}{\partial b_3}$	1.4894e-10	1.4894e-10	1.4894e-10	1.4894e-10

Table 2. 3 layer network: Gradient computation error between analytical and numerical methods for batch size 100

ε	0.00001	0.0001	0.001	0.01
$\frac{\partial J}{\partial W_1}$	4.8902e-06	4.8902e-07	4.8902e-08	4.8902e-09
$\frac{\partial J}{\partial W_2}$	3.8604e-06	3.8604e-07	3.8604e-08	3.8604e-09
$\frac{\partial J}{\partial W_3}$	4.0655e-06	4.0655e-07	4.0655e-08	4.0655e-09
$\frac{\partial J}{\partial W_4}$	3.1165e-09	3.1165e-09	3.1165e-09	2.4072e-09
$\frac{\partial J}{\partial b_1}$	3.4149e-09	3.4149e-09	3.4149e-09	5.5863e-10
$\frac{\partial J}{\partial b_2}$	1.3143e-09	1.3143e-09	1.3143e-09	1.3143e-09
$\frac{\partial J}{\partial b_3}$	2.0237e-09	2.0237e-09	2.0237e-09	2.0237e-09
$\frac{\partial J}{\partial b_4}$	1.0984e-10	1.0984e-10	1.0984e-10	1.0984e-10

Table 3. 4 layer network: Gradient computation error between analytical and numerical methods for batch size 100

2) Training and testing of a K-layer network:

- Results achieved for a 2-layer network: The results are same as in Assignment 2

Regularization (λ)	Step size (n_s)	Cycles	Training Accuracy (%)		Validation Accuracy (%)		Test Accuracy (%)
			Mean	At the end of the cycles	Mean	At the end of the cycles	
0	500	1	48.2364	66.02	37.1891	44.85	44.96
0.01	500	1	46.1500	60.72	37.5155	45.53	45.88
0.01	800	3	57.6012	71.68	41.7631	46.71	46.77

- *Results of 3 layer network:*
  For parameters provided in the assignment, with random shuffling of data after each epoch:

λ	Step size (n_s)	Cycles	Batch size	Eta_min	Eta_max	Test accuracy (%)
0.005	$5 * (\frac{45000}{batch\ size}) =$ 2250	2	100	1e-5	1e-1	52.94

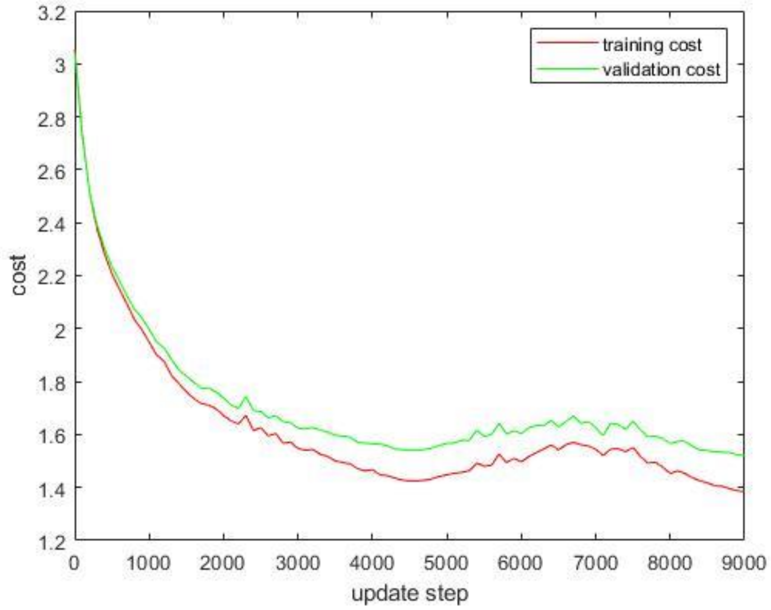

Fig 1. Training and validation cost (layers = 3, lambda = 0.005, batch = 100, cycle = 2, update_steps = 2250, Eta_min = 1e-5, Eta_max = 1e-1)

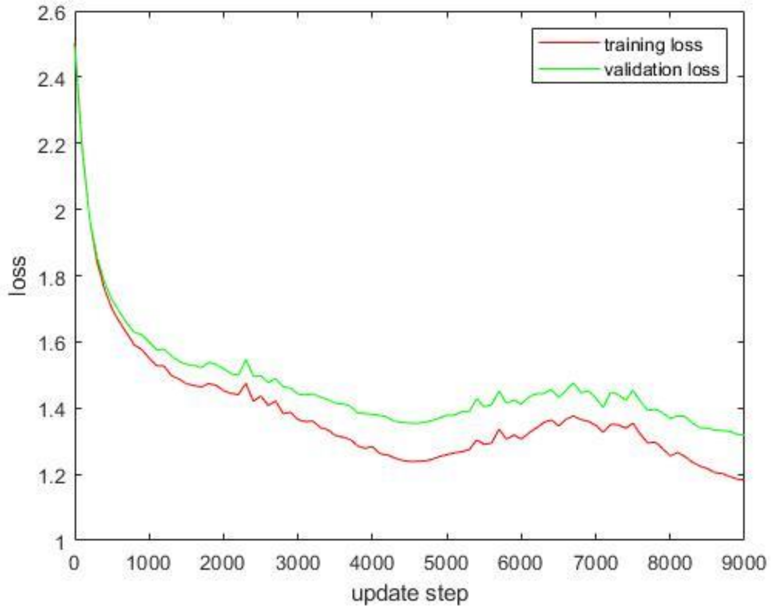

Fig 2. Training and validation loss (lambda = 0.005, batch = 100, cycle = 2, update_steps = 2250, Eta_min = 1e-5, Eta_max = 1e-1)

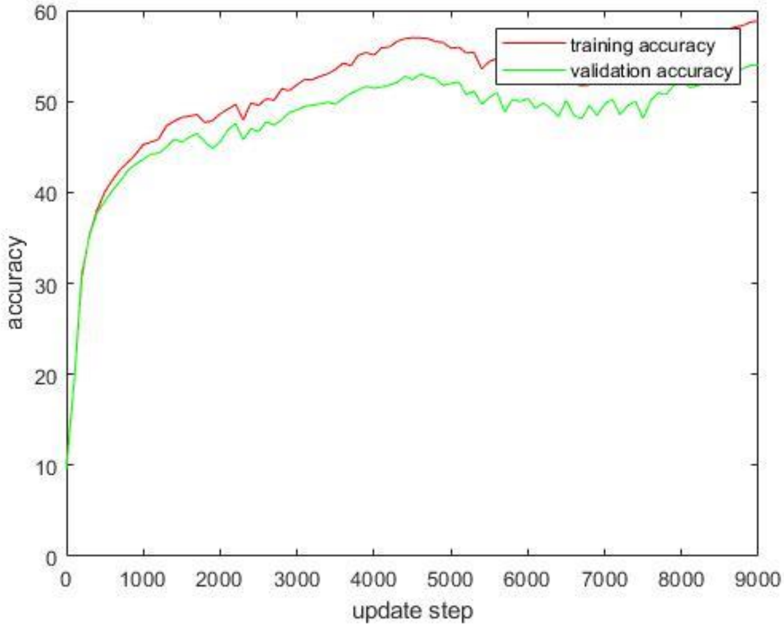

Fig 3. Training and validation accuracy (lambda = 0.005, batch = 100, cycle = 2, update_steps = 2250, Eta_min = 1e-5, Eta_max = 1e-1)

- *Results of 9 layer network:*

For parameters provided in the assignment, with random shuffling of data after each epoch:

λ	Step size (n_s)	Cycles	Batch size	Eta_min	Eta_max	Test accuracy (%)
0.005	$5 * (\frac{45000}{batch\ size}) =$ 2250	2	100	1e-5	1e-1	41.51

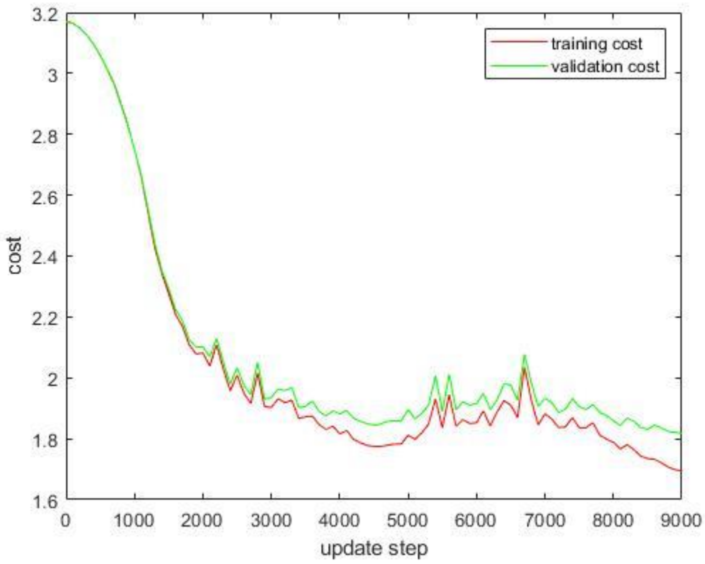

Fig 4. Training and validation cost (layers = 9, lambda = 0.005, batch = 100, cycle = 2, update_steps = 2250, Eta_min = 1e-5, Eta_max = 1e-1)

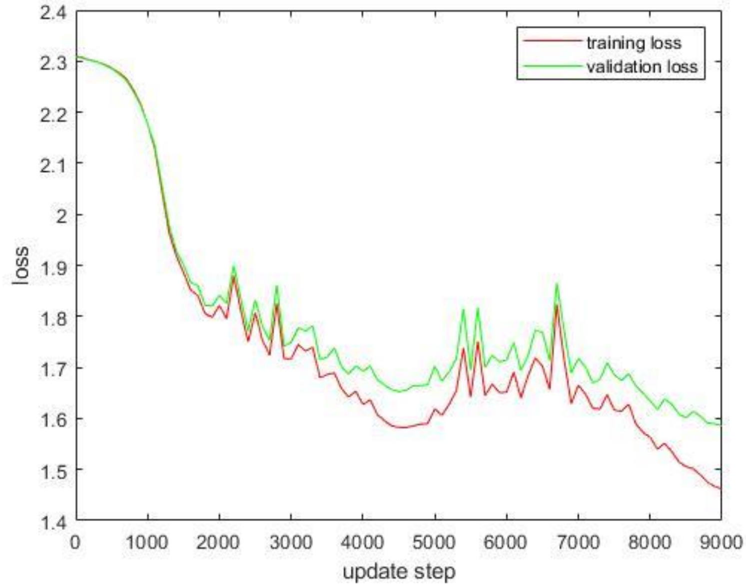

Fig 5. Training and validation loss (layers = 9, lambda = 0.005, batch = 100, cycle = 2, update_steps = 2250, Eta_min = 1e-5, Eta_max = 1e-1)

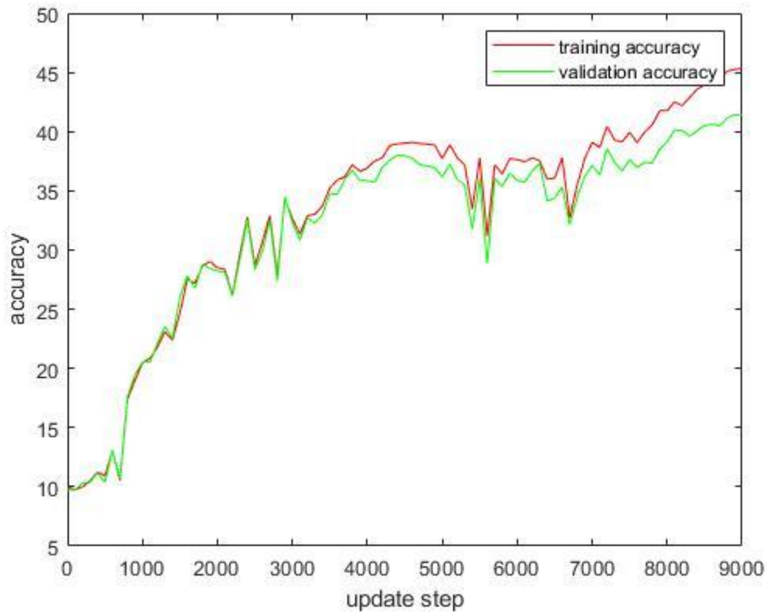

Fig 6. Training and validation accuracy (layers = 9, lambda = 0.005, batch = 100, cycle = 2, update_steps = 2250, Eta_min = 1e-5, Eta_max = 1e-1)

K- layer network implementation with Batch Normalisation:

1) Gradient computation check:

Analytical gradient computation was compared against numerical gradient computation for different batch sizes and ε values. The error results are presented in tables 3 and 4. The error values are considerably small. Hence, we can conclude that the analytical method works well.

ε	0.00001	0.0001	0.001	0.01
$\frac{\partial J}{\partial W_1}$	1.3873e-05	1.8887e-06	1.8887e-06	1.8887e-06
$\frac{\partial J}{\partial W_2}$	8.1528e-07	6.4747e-07	9.4057e-08	1.0090e-08
$\frac{\partial J}{\partial \gamma_1}$	6.3210e-07	6.3210e-07	7.5585e-08	7.5585e-08
$\frac{\partial J}{\partial b_1}$	1.2212e-12	1.2212e-13	1.2212e-14	1.2212e-15
$\frac{\partial J}{\partial b_2}$	4.4104e-07	4.4104e-07	1.0180e-07	1.0180e-07
$\frac{\partial J}{\partial \beta_1}$	1.3216e-07	1.3216e-07	6.2095e-08	1.1225e-08

Table 3. 2 layer network: Gradient computation error between analytical and numerical methods for batch size 100

ε	0.00001	0.0001	0.001	0.01
$\frac{\partial J}{\partial W_1}$	2.1667e-05	2.2073e-06	3.1169e-07	3.1169e-08
$\frac{\partial J}{\partial W_2}$	9.5766e-06	1.6820e-06	1.6820e-07	1.6820e-08
$\frac{\partial J}{\partial W_3}$	4.5814e-07	4.5814e-07	1.5078e-07	1.5078e-08
$\frac{\partial J}{\partial \gamma_1}$	5.6487e-07	5.6487e-07	1.1941e-07	1.1941e-08
$\frac{\partial J}{\partial \gamma_2}$	6.8074e-07	6.8074e-07	1.1143e-07	1.1143e-08
$\frac{\partial J}{\partial b_1}$	1.2876e-12	1.2876e-13	1.2876e-14	1.2876e-15
$\frac{\partial J}{\partial b_2}$	3.7401e-12	3.7401e-13	3.7401e-14	3.7401e-15
$\frac{\partial J}{\partial b_3}$	1.1814e-08	1.1814e-08	1.1814e-08	1.0306e-08
$\frac{\partial J}{\partial \beta_1}$	1.2584e-07	1.2584e-07	1.1362e-07	1.2797e-08
$\frac{\partial J}{\partial \beta_2}$	2.5919e-07	2.5919e-07	3.7120e-08	6.1645e-09

Table 4. 3 layer network: Gradient computation error between analytical and numerical methods for batch size 100

2) **Training and testing of K- layer network with Batch Normalization (with random shuffling of data)**

- *Results of 3 layer network with Batch Normalization:*

λ	Step size (n_s)	Cycles	Batch size	Eta_min	Eta_max	Test accuracy (%)
0.005	$5 * (\frac{45000}{batch\ size}) =$ 2250	2	100	1e-5	1e-1	53.37

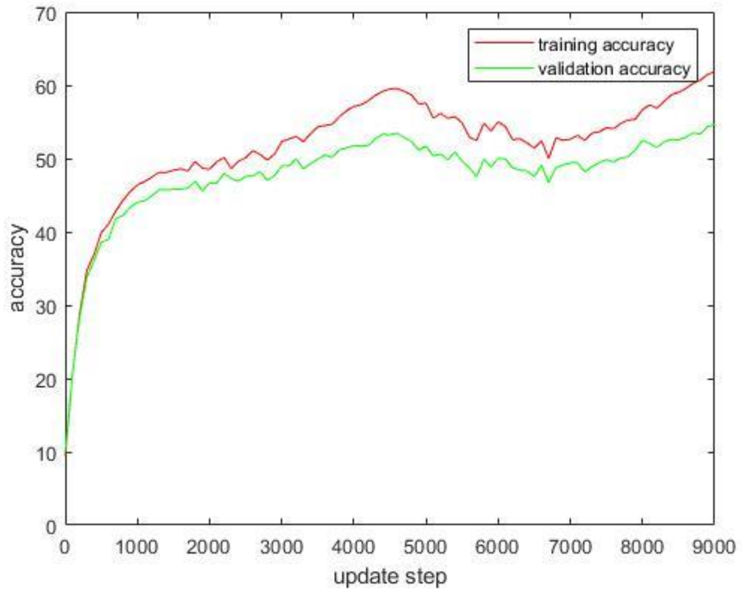

Fig 7. Training and validation accuracy with Batch Normalizaion (layers = 3, lambda = 0.005, batch = 100, cycle = 2, update_steps = 2250, Eta_min = 1e-5, Eta_max = 1e-1)

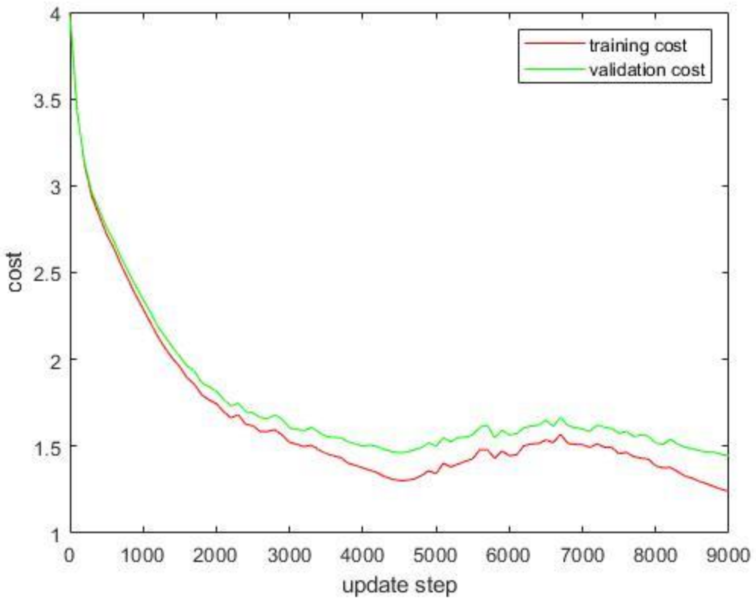

Fig 8. Training and validation cost with Batch Normalizaion (layers = 3, lambda = 0.005, batch = 100, cycle = 2, update_steps = 2250, Eta_min = 1e-5, Eta_max = 1e-1)

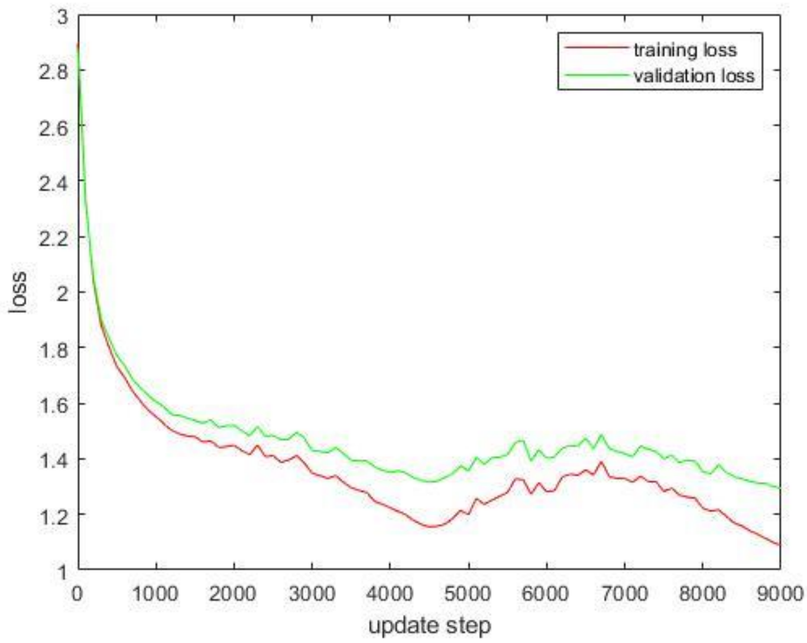

Fig 9. Training and validation loss with Batch Normalizaion (layers = 3, lambda = 0.005, batch = 100, cycle = 2, update_steps = 2250, Eta_min = 1e-5, Eta_max = 1e-1)

3) Search for regularization (λ) – 3 Layer network

- Coarse search: Unigrid (-5, 0.5, -1)

λ_{min}	λ_{max}	λ	Validation accuracy (%)
$10^{-3.5}$	10^{-1}	0.00753375943671716	55.8
$10^{-2.5}$	10^{-2}	0.00328968767500303	55.54
$10^{-2.5}$	$10^{-1.5}$	0.00420245980441812	55.3

Table 5. Three best validation accuracies obtained for coarse search of lambda in the range of $[10^{-5}, 10^{-1}]$

- Fine search : Unigrid (-3, 0.2, -1)

λ_{min}	λ_{max}	λ	Validation accuracy (%)
10^{-3}	$10^{-1.8}$	0.00455344869218005	55.88
$10^{-2.4}$	$10^{-1.4}$	0.00529058343982553	55.82
$10^{-2.6}$	$10^{-2.2}$	0.00513742653471657	55.76

Table 6. Three best validation accuracies obtained for coarse search of lambda in the range of $[10^{-3}, 10^{-1}]$

From the searches, lambda value is chosen as 0.0046

4) Training and testing using the chosen λ value with Batch normalization:

- *Results of 3 layer network with Batch Normalization:*

λ	Step size (n_s)	Cycles	Batch size	Eta_min	Eta_max	Test accuracy (%)
0.0046	$5 * (\frac{45000}{batch\ size}) = 2250$	3	100	1e-5	1e-1	53.49

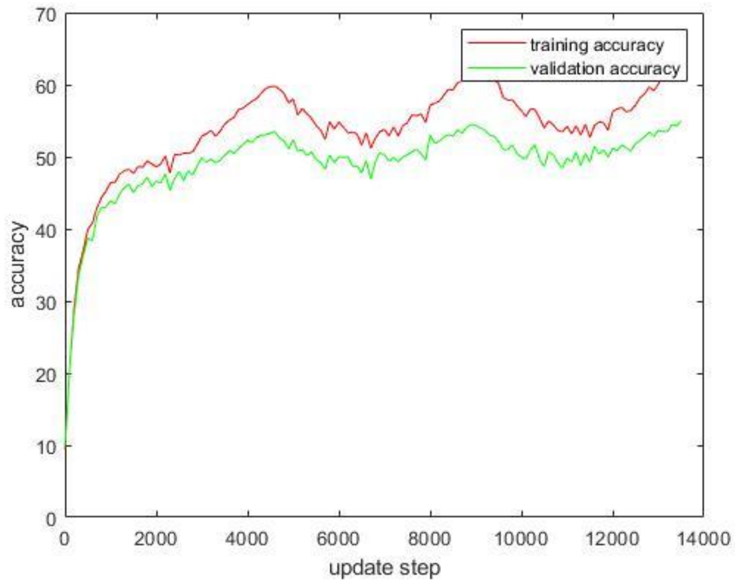

Fig 10. Training and validation accuracy with Batch Normalization (layers = 3, lambda = 0.0046, batch = 100, cycle = 3, update_steps = 2250, Eta_min = 1e-5, Eta_max = 1e-1)

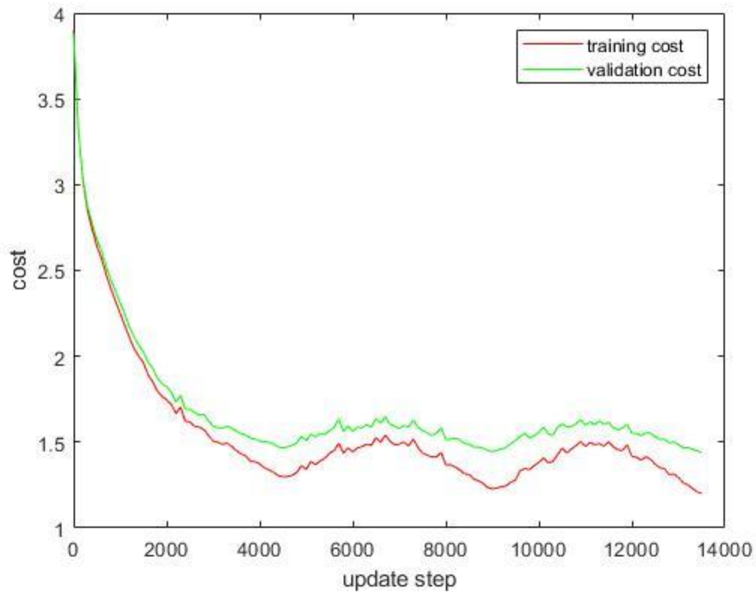

Fig 11. Training and validation cost with Batch Normalization (layers = 3, lambda = 0.0046, batch = 100, cycle = 3, update_steps = 2250, Eta_min = 1e-5, Eta_max = 1e-1)

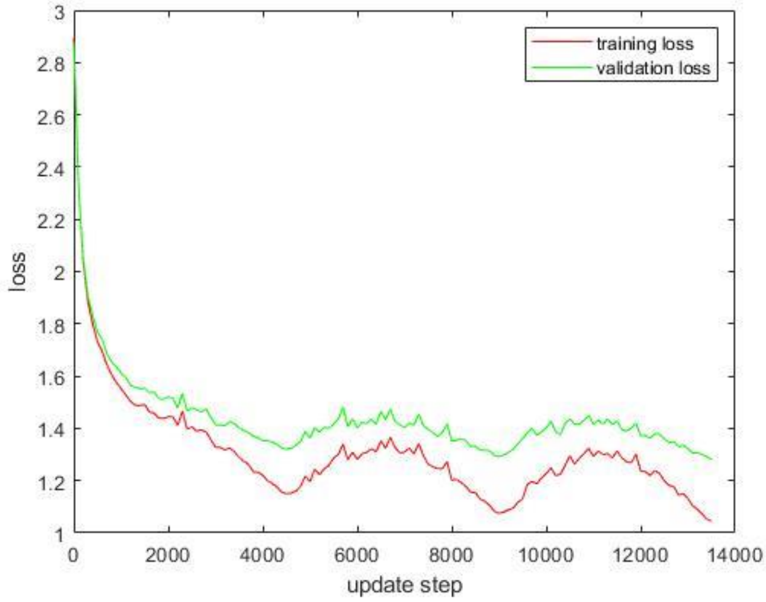

Fig 12. Training and validation loss with Batch Normalization (layers = 3, lambda = 0.0046, batch = 100, cycle = 3, update_steps = 2250, Eta_min = 1e-5, Eta_max = 1e-1)

- *Results of 9 layer network with Batch Normalization:*
  The results obtained using batch normalization for a deep 9 layer network is much better when compared to the results obtained from not using batch normalization, which was (41.51%)

λ	Step size (n_s)	Cycles	Batch size	Eta_min	Eta_max	Test accuracy (%)
0.0046	$5 * \left(\frac{45000}{batch\ size}\right) =$ 2250	3	100	1e-5	1e-1	51.85

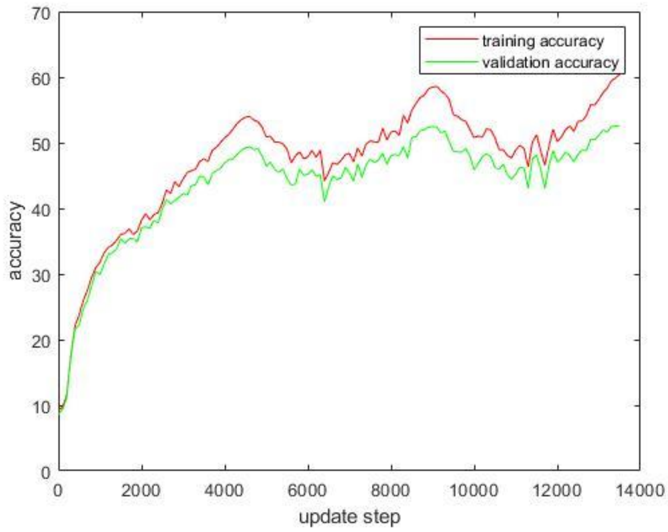

Fig 13. Training and validation accuracy with Batch Normalization (layers = 9, lambda = 0.0046, batch = 100, cycle = 3, update_steps = 2250, Eta_min = 1e-5, Eta_max= 1e-1)

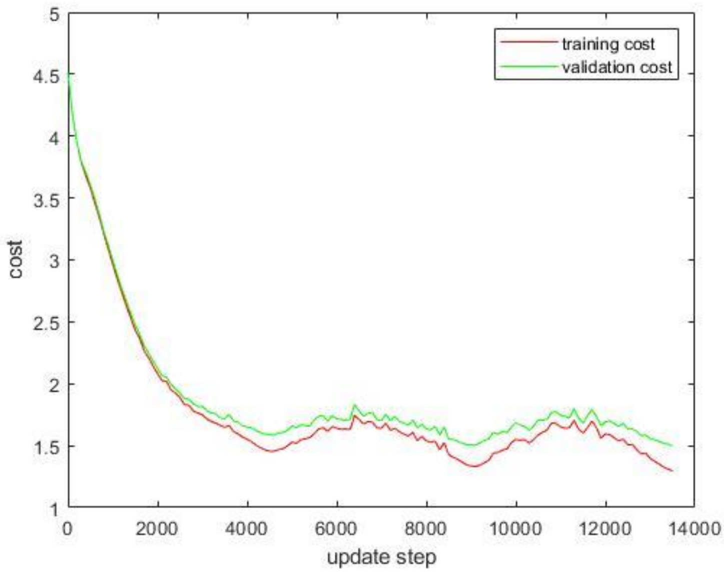

Fig 14. Training and validation cost with Batch Normalization (layers = 9, lambda = 0.0046, batch = 100, cycle = 3, update_steps = 2250, Eta_min = 1e-5, Eta_max = 1e-1)

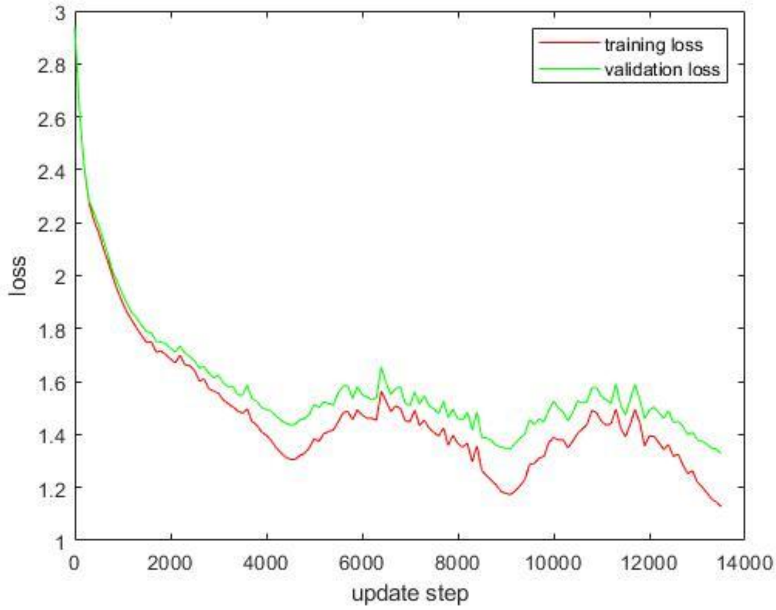

Fig 15. Training and validation loss with Batch Normalization (layers = 9, lambda = 0.0046, batch = 100, cycle = 3, update_steps = 2250, Eta_min = 1e-5, Eta_max = 1e-1)

Sensitivity to initialization:

A 9-layer network with initial values of the network parameters drawn from a normal distribution having standard deviation: 1e-1, 1e-3, 1e-4. Through this experiment it is found that implementing a K-layer network with Batch Normalization results in a network performance which is less sensitive to initialization. From figure 17, it can be seen that the network loss remains almost constant.

Network parameters are:

eta_min = 1e-5;
eta_max = 1e-1;
n_batch = 100;
cycles = 3;
updateStep = 5 *(45000/n_batch);

- With Batch Normalization

Standard deviation	Test Accuracy	Max. training accuracy	Max. validation accuracy
1e-1	52.62	60.0778	53.4600
1e-3	51.87	59.3867	52.7400
1e-4	51.37	59.611	51.86

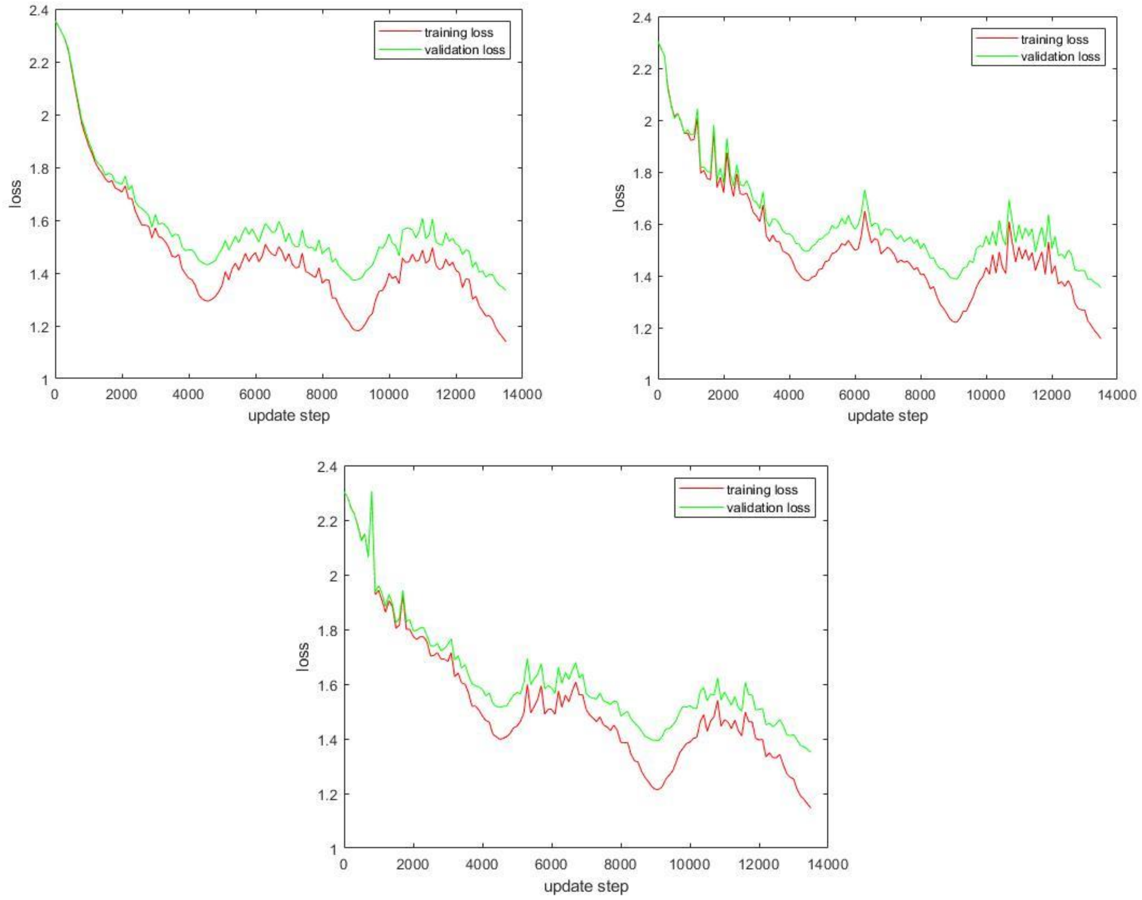

Fig 16. With batch normalization- Training and validation loss evolution for initialization with standard deviation: 1e-1 (top left), 1e-3 (top right), 1e-4 (bottom)

- Without Batch Normalization

Standard deviation	Test Accuracy	Max. training accuracy	Max. validation accuracy
1e-1	10	10.0556	10.6400
1e-3	10	10.0556	10.6400
1e-4	10	10.0556	10.6400

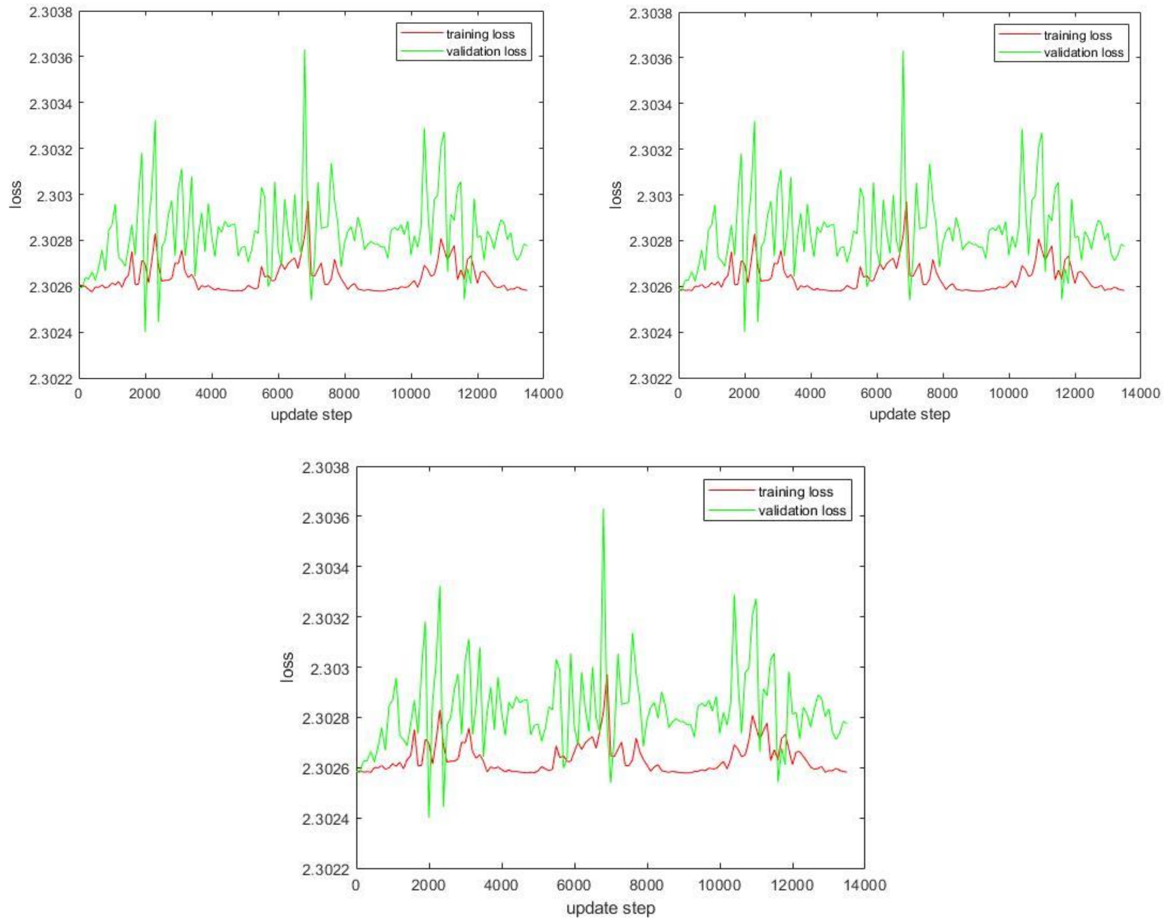

Fig 17. Without batch normalization- Training and validation loss evolution for initialization with standard deviation: 1e-1 (top left), 1e-3 (top right), 1e-4 (bottom)