Homework Report: Assignment 3 k-layer network with batch normalization for multi-label classification /FDD3424

Yuning Wang

KTH Royal Institute of Technology
SCI, Engineering Mechanics

May 6, 2024

1 Validating the computation of gradient.

In this section we report the debugging of the gradient computation by comparing our analytical results to with the one yielded by numerical schemes. We compute the gradient compute using a single image and compare the results from the numerical methods based on *central difference* methods, namely **ComputeGradsNumSlow**. We assess the relative error ε as:

$$\varepsilon = \frac{|g_a - g_n|}{\max(|g_a|, |g_n|)} \tag{1}$$

where $\xi = 1 \times 10^{-5}$ is the numerical disretization for computing the gradient whereas g_a and g_n denotes the gradient from analytical and numerical methods, respectively.

We carried out the validation on a 4-layer network for the case with and without employing batch normalization (BN). Note that we truncate the input data dimension from $d = 3072$ to 10 for a single sample of image. Tab. 1 summarizes the obtained ε between analytical and numerical solution, where all the obtained results are smaller than the $1 \times 10{-}4$, indicating the implementation in this report on gradient descent is reliable. And it is worth noting that regarding the network without employing BN, the relative error increases as the gradient back propagates to the layer closer to the input.

Parameter	*No BN*	*BN*
ε_{W_1}	9.62×10^{-8}	0.0
ε_{b_1}	9.02×10^{-9}	0.0
ε_{W_2}	2.07×10^{-7}	0.0
ε_{b_2}	5.93×10^{-9}	0.0
ε_{W_3}	4.68×10^{-7}	0.0
ε_{b_3}	4.33×10^{-10}	0.0
ε_{W_3}	8.25×10^{-8}	0.0
ε_{b_3}	2.63×10^{-10}	8.23×10^{-11}

Table 1: Obtained relative error by comparing the implemented analytical solution and existing numerical methods.

2 The effect of batch normalization on k-layer network

In this section we report the results of so-called the ablation studies of employing the BN on the k-layer with the same architecture. We consider a 3-layer network with 50 and 50 nodes in the first and second hidden layer respectively with the same learning parameters. And we keep the same amount of the parameters in architecture but increase the number of layers to 9. The 9-layer network comprises by 8 hidden layers using number of nodes are 50, 30, 20, 20, 10, 10, 10, 10, respectively. For mini-batch gradient descent, we adopt hyper-parameter setting: n_batch=100, $eta\_min = 1 \times 10^{-5}$, $eta\_max = 1 \times 10^{-1}$, $\lambda = 0.005$, 2 cycles of training and n_s = 5 * 45,000 / n_batch. We adopt $\alpha = 0.8$ to initialize the moving average in BN, and we employ He initialization for the weight matrices with $\sigma = 1 \times 10^{-1}$. Note that 5,000 samples of data are used as validation data during training.

No. layers	*without BN*	*BN*
3	52.95%	53.14%
9	44.02%	51.75%

Table 2: Achieved accuracy on test dataset by the proposed networks in this section. The highest accuracy achieved for each architecture is highlighted by red.

Tab. 2 summaries the obtained accuracy on the test dataset for all the cases mentioned above. For the model not using the batch normalization, stacking more layers leads to more over-fitting thus yield a lower testing accuracy, decreased by 17%. However, for the models employed batch normalization, the accuracy also downgrades after stacking more layers but just by nearly 1%. The results imply that the BN improves the accuracy of the mini-batch gradient descent for a larger architecture.

Fig. 1 and 2 depicts the training curves obtained from the training the 3-layer and 9-layer networks using and without using the batch normalization, respectively. One can observe that the models with BN converges to a lower loss magnitude (i.e the elbow appears earlier in the loss evolution.) quicker than the model without BN, which indicates the BN facilities the stability of mini-batch gradient descent.

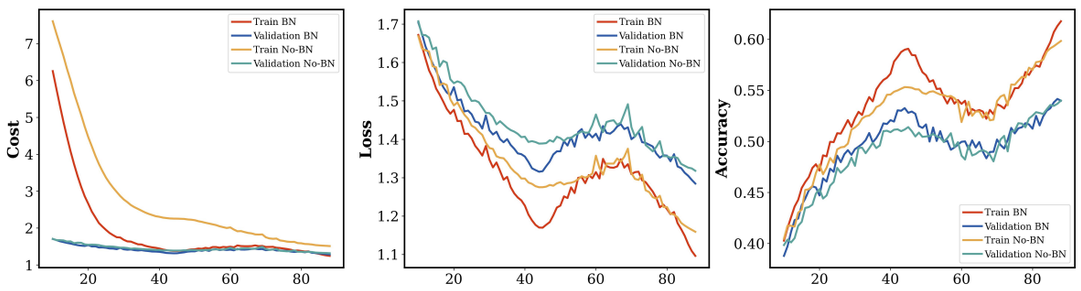


Figure 1: Training curves of 3-layer network with 50 and 50 nodes in the first and second hidden layer respectively. The red and blue line denotes the training and validation curves obtained by model using BN, while the yellow and green line denotes the training and validation curves obtained by model without BN, respectively. The parameter n_batch=100, $eta\_min = 1 \times 10^{-5}$, $eta\_max = 1 \times 10^{-1}$, $\lambda = 0.005$, 2 cycles of training and $n\_s = 5$ * 45,000 / n_batch are adpot.

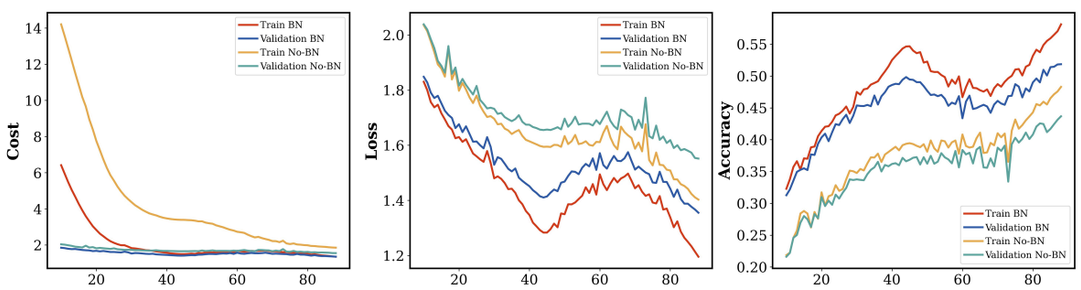


Figure 2: Training curves of 9-layer network with 50 and 50 nodes in the first and second hidden layer respectively. The red and blue line denotes the training and validation curves obtained by model using BN, while the yellow and green line denotes the training and validation curves obtained by model without BN, respectively. The parameter n_batch=100, $eta\_min = 1 \times 10^{-5}$, $eta\_max = 1 \times 10^{-1}$, $\lambda = 0.005$, 2 cycles of training and $n\_s = 5$ * 45,000 / n_batch are adpot.

3 Grid search for optimal parameter

In this section, we report the grid search of the optimal regularization λ for a 3-layer network trained with BN. We adopt $\alpha = 0.8$ to initialize the moving average in BN, and we employ He initialization for the weight matrices with $\sigma = 1 \times 10^{-1}$. Note that 5,000 samples of data are used as validation data during training.

We first conduct a coarse search on 8 samples randomly selected from a range between $1 \times 10^{-5}, 1 \times 10^{-1}$ and summarize the test performing in Tab. 3, where the $\lambda = 4.24 \times 10^{-3}$ yields the best testing accuracy of 53.8%.

λ	*Test Acc (%)*
1.233×10^{-5}	51.02
1.671×10^{-4}	51.19
7.570×10^{-4}	51.95
4.237×10^{-3}	53.84
4.731×10^{-3}	53.43
9.889×10^{-3}	53.23
1.960×10^{-2}	52.78
5.413×10^{-2}	50.52

Table 3: Achieved accuracy on test dataset by the proposed networks in this section. The highest accuracy achieved for each architecture is highlighted in red.

Subsequently, we narrow the range for search between 1×10^{-3} to 1×10^{-2} and randomly sampled another 8 values for λ for a finer grid search. Tab 4 summarizes the testing accuracy achieved for the sampled values, where $\lambda = 3.246 \times 10^{-3}$ yields the highest accuracy of 53.70%. Therefore, our results indicate that the $\lambda = 3.246 \times 10^{-3}$ is the optimal regularization term to adpot in the cost function in this study.

λ	*Test Acc (%)*
3.246×10^{-3}	54.15
4.496×10^{-3}	53.70
5.431×10^{-3}	53.82
6.736×10^{-3}	53.52
6.829×10^{-3}	53.80
7.488×10^{-3}	53.78
8.157×10^{-3}	52.79
9.261×10^{-3}	53.87

Table 4: Achieved accuracy on test dataset by the proposed networks in this section. The highest accuracy achieved for each architecture is highlighted in red.

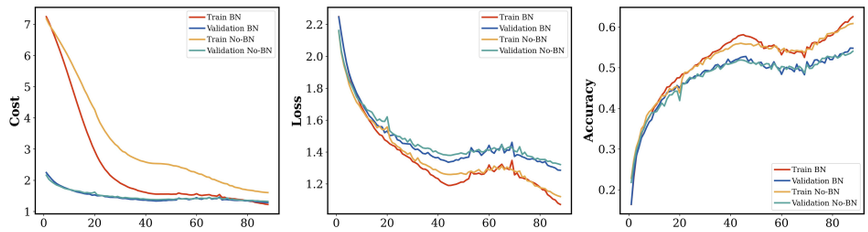
4 Sensitivity studies on He initialization

At last, we employ the 3-layer network trained with and without BN to investigate the sensitivity of initialization. We employ the He initialization which uses zero for mean (μ) and a constant value for standard deviation (σ) for the normal distribution, and sample parameters in the weight matrices from this distribution as an initial guess.

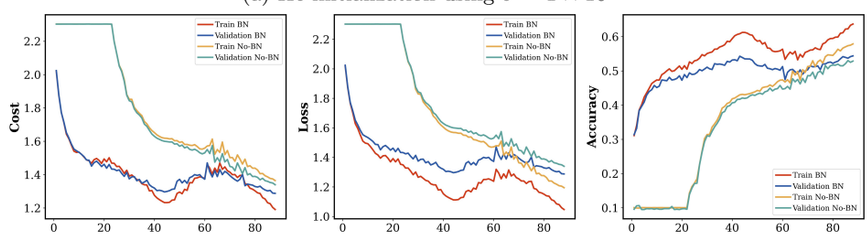
We consider $\sigma = 1 \times 10^{-4}$, 1×10^{-3} and 1×10^{-1} in the present study. Note that we adopt the optimal $\lambda = 3.246 \times 10^{-3}$ for all the network, whereas the hyper parameters for training are the same as we reported for the previous sections.

σ	*without BN*	*with BN*
1×10^{-1}	53.24%	54.15%
1×10^{-3}	51.75%	53.21%
1×10^{-4}	10%	53.00%

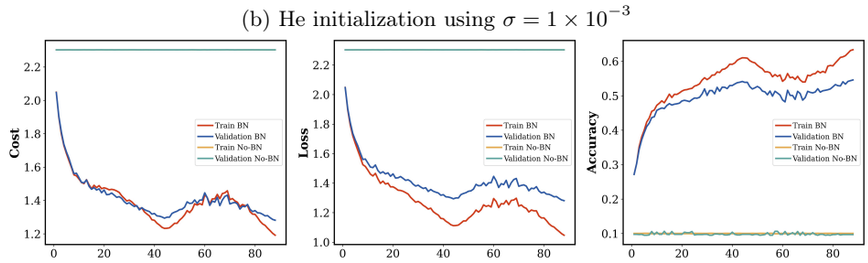
Table 5: Achieved accuracy on test dataset (in percentage) by the proposed networks in this section. The highest accuracy achieved for each architecture is highlighted in red.



(a) He initialization using $\sigma = 1 \times 10^{-1}$



(b) He initialization using $\sigma = 1 \times 10^{-3}$



(c) He initialization using $\sigma = 1 \times 10^{-4}$

Figure 3: Training curves of 3-layer network with 50 and 50 nodes in the first and second hidden layer respectively. The red and blue line denotes the training and validation curves obtained by model using BN, while the yellow and green line denotes the training and validation curves obtained by model without BN, respectively. The parameter n_batch=100, $eta\_min = 1 \times 10^{-5}$, $eta\_max = 1 \times 10^{-1}$, $\lambda = 0.005$, 2 cycles of training and $n\_s = 5$ * 45,000 / n_batch are adpot.

Tab .5 summarizes the obtained testing accuarcy for the network using and not using the BN during training. The obtained accuracy for the models using BN exhibit a slight decrease when suppressing the σ to 1×10^{-4}. However, it is worth noting that the performance of networks without BN drastically downgrades from 53.24% to 10% as σ decrease from 1×10^{-1} to 1×10^{-4}. Moreover, the transient of performance can be observed in the training curves of the networks. One can observed that when the σ decreases, the training curves of the network without using BN converge at a very high value for the cost and loss function, indicating the mini-batch gradient descent have stuck at local minima at very beginning of training stage. The results indicate that BN reduces the sensitivity of initialization for the network, facilitating the stability of mini-batch gradient descent.