

INDIAN INSTITUTE OF TECHNOLOGY, JODHPUR

Deep Learning Minor 2

Minor REPORT

Submitted By:

Rutvik Jethava (M22CS011)



॥ त्वं ज्ञानमयो विज्ञानमयोऽसि ॥

Question 1:

Dataset details

- TinyImageNet dataset

Weight Initialization:

- Xavier initialization

Pooling Operation Details:

- AvgPool

Model Details:

- For M.Tech/PhD students:

Teacher Network layers:

- your network should have 12 conv layers and 1 pool layer.
- your network will have 12 filters in the first layer.

Student Network:


- then student network is 3 conv layers + 1 pool layer

Fully-Connected layers (in both teacher and student networks):

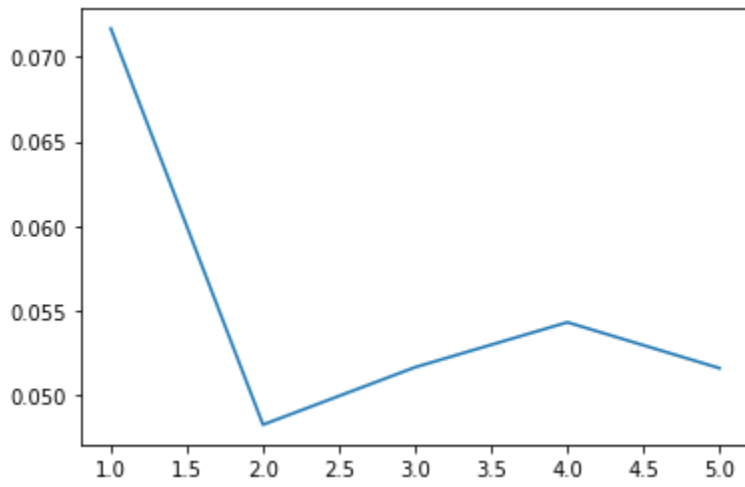
- your network should have 1 FC layer with 1024 nodes.

Teacher model accuracy without EMA

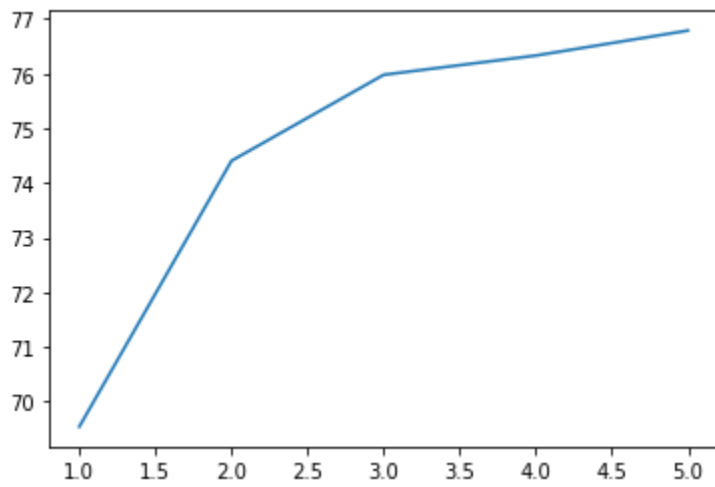
Training Accuracy & Test Accuracy

```
* 100%  5/5 [07:33<00:00, 90.55s/it]
Epoch: 0/5 | step:1562/1563 | Loss:0.0717 | Accuracy 69.54
Accuracy : 0.0
Epoch: 1/5 | step:1562/1563 | Loss:0.0483 | Accuracy 74.41
Accuracy : 0.0
Epoch: 2/5 | step:1562/1563 | Loss:0.0517 | Accuracy 75.98
Accuracy : 0.0
Epoch: 3/5 | step:1562/1563 | Loss:0.0543 | Accuracy 76.33
Accuracy : 0.0
Epoch: 4/5 | step:1562/1563 | Loss:0.0516 | Accuracy 76.79
Accuracy : 0.0
```

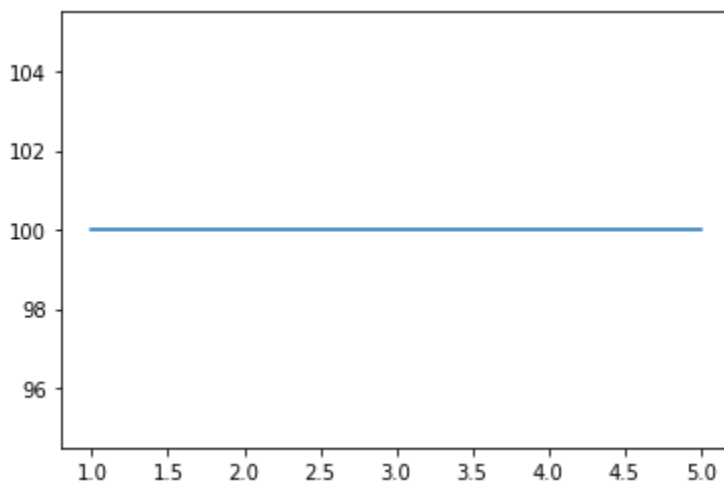
Epoch Vs Train Loss



Epoch vs Accuracy



Epoch vs Test Accuracy



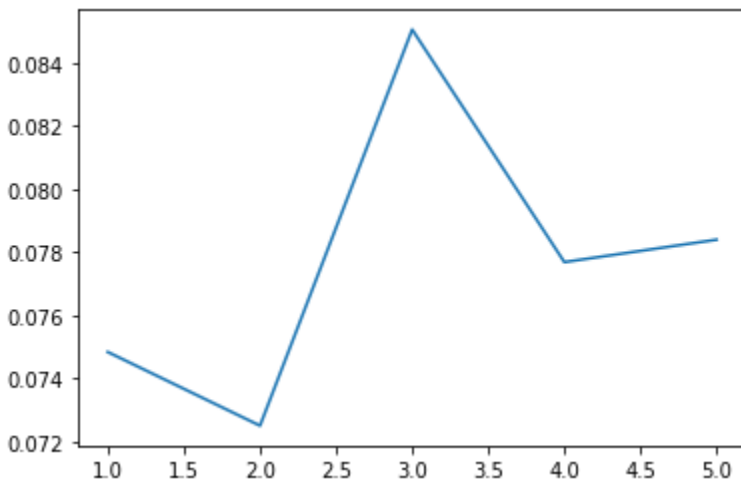
Student without EMA model accuracy

Train Accuracy Vs Test Accuracy

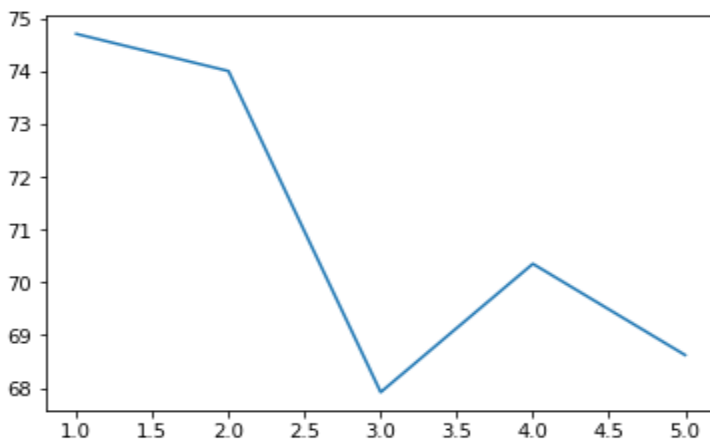
```
100% ██████████ 5/5 [09:05<00:00, 108.83s/it]
/usr/local/lib/python3.9/dist-packages/torch/nn/functional.py:2916: UserWarning: reduction: 'mean' not implemented for 'Half'
  warnings.warn(
Epoch: 0/5 | step:1562/1563 | Loss:nan | Accuracy 0.44
Test : Epoch 1: Loss = 0.000, Test Accuracy = 100.000 %
Epoch: 1/5 | step:1562/1563 | Loss:nan | Accuracy 0.50
Test : Epoch 2: Loss = 0.000, Test Accuracy = 100.000 %
Epoch: 2/5 | step:1562/1563 | Loss:nan | Accuracy 0.50
Test : Epoch 3: Loss = 0.000, Test Accuracy = 100.000 %
Epoch: 3/5 | step:1562/1563 | Loss:nan | Accuracy 0.50
Test : Epoch 4: Loss = 0.000, Test Accuracy = 100.000 %
Epoch: 4/5 | step:1562/1563 | Loss:nan | Accuracy 0.50
Test : Epoch 5: Loss = 0.000, Test Accuracy = 100.000 %
```

Teacher model accuracy EMA


Epoch Vs Train Loss



Epoch vs Accuracy



Student model accuracy EMA

100%  5/5 [09:30<00:00, 114.28s/it]

Train Epoch: 0/5		step:1562/1563		Loss:nan		Accuracy 0.50
Test Epoch: 0/5		step:1562/1563		Loss:nan		Accuracy 100.00
Train Epoch: 1/5		step:1562/1563		Loss:nan		Accuracy 0.50
Test Epoch: 1/5		step:1562/1563		Loss:nan		Accuracy 100.00
Train Epoch: 2/5		step:1562/1563		Loss:nan		Accuracy 0.50
Test Epoch: 2/5		step:1562/1563		Loss:nan		Accuracy 100.00
Train Epoch: 3/5		step:1562/1563		Loss:nan		Accuracy 0.50
Test Epoch: 3/5		step:1562/1563		Loss:nan		Accuracy 100.00
Train Epoch: 4/5		step:1562/1563		Loss:nan		Accuracy 0.50
Test Epoch: 4/5		step:1562/1563		Loss:nan		Accuracy 100.00

Question 2:

Question 2.1: How to fix the constant validation accuracy in CNN model training? is that a vanishing gradient problem?

ChatGPT Ans:

If the validation accuracy of a Convolutional Neural Network (CNN) model remains constant during training, it could be a sign of several issues, one of which could be vanishing gradients. However, it is not the only reason for constant validation accuracy. Here are some possible reasons and solutions to this problem:

1. **Poor choice of hyperparameters:** The learning rate, batch size, and other hyperparameters can have a significant impact on the training process. A poorly chosen set of hyperparameters can lead to the model getting stuck at a suboptimal solution. In such cases, you may need to experiment with different hyperparameters to find the optimal values.
2. **Insufficient training data:** If the training data is too small or not representative of the population, the model may fail to generalize well to new data. In such cases, you may need to acquire more data or use data augmentation techniques to increase the size of the training set.
3. **Overfitting:** If the model is overfitting the training data, it may perform well on the training set but poorly on the validation set. In such cases, you may need to apply regularization techniques such as dropout, weight decay, or early stopping to prevent overfitting.
4. **Vanishing gradients:** If the validation accuracy remains constant, it could be a sign of vanishing gradients. In such cases, you may need to use a different initialization strategy or modify the architecture of the CNN model. You can also try using gradient clipping, which limits the size of the gradients during backpropagation.
5. **Bug in the code:** Sometimes, a constant validation accuracy can be caused by a bug in the code. You may need to review your code and check for any errors or mistakes.

In summary, the constant validation accuracy problem in CNN model training could be caused by a range of issues, including vanishing gradients. To diagnose and fix the issue, it is important to understand the specific cause and take appropriate measures to address it.

Actual Ans:

When training a neural network, constant validation accuracy is a common issue because the network simply remembers the sample, resulting in an overfitting issue. Overfitting a model indicates that the neural network model performs exceptionally well on the training sample, but its performance decreases on the validation set. Here are some suggestions for improving CNN's constant validation accuracy –

- The dataset should always be divided into training, validation, and test sets.
- Changing the parameters of the neural network through trial and error can help address this issue when working with limited data.
- Increasing the training dataset's capacity.
- Implement batch normalisation.
- Regularization

Question 2.2: Suppose you have a dataset with 1000 observations and 10 features, and you want to train a linear regression model using L1 and L2 regularization. You split the data into a training set of 800 observations and a validation set of 200 observations. You set the regularization strength hyperparameters alpha and lambda to 0.01.

What is the cost function for the regularized linear regression model?

ChatGPTAns:

The cost function for the regularized linear regression model with L1 and L2 regularization is given by:

$$\text{Cost} = (1/2m) * [\text{sum}((h(x_i) - y_i)^2) + \text{lambda} * \text{sum}(\text{abs}(\text{theta})) + \text{alpha} * \text{sum}(\text{theta}^2)]$$

where m is the number of training examples, $h(x_i)$ is the predicted value for the i-th example, y_i is the true value for the i-th example, theta is the vector of coefficients, lambda and alpha are the regularization strengths for L1 and L2 regularization, respectively.

My ans almost the same answer as above.

Question 2.3: Given a Convolution Layer with 8 filters, a filter size of 6, a stride of 2, and a padding of 1. For an input feature map of $32 \times 32 \times 32$, what is the output dimensionality after applying the Convolution Layer to the input?

ChatGPT Ans:

The output spatial dimensionality is given by the formula:

$$O = (I - K + 2P) / S + 1$$

where I is the input spatial dimensionality, K is the filter size, P is the padding, and S is the stride.

In this case, the input feature map has a spatial dimensionality of 32×32 , and a depth of 32. The filter size is 6, the stride is 2, and the padding is 1.

So, for each of the 8 filters, the output spatial dimensionality is:

$$O = (32 - 6 + 2(1)) / 2 + 1 = 15$$

Therefore, the output spatial dimensionality after applying the convolution layer is 15×15 , and the output depth is 8 (equal to the number of filters).

So the output dimensionality after applying the Convolution Layer to the input is $15 \times 15 \times 8$.

Ans:

Actual ans is $15 * 15 * 8$

Referred Websites:

<https://towardsdatascience.com/pytorch-ignite-classifying-tiny-imagenet-with-efficientnet-e5b1768e5e8f>

<https://discuss.pytorch.org/t/knowledge-distillation/163807>

<https://abdulkaderhelwan.medium.com/implementation-of-teacher-student-model-in-pytorch-a1195734dd08>

<https://medium.com/analytics-vidhya/knowledge-distillation-in-a-deep-neural-network-c9dd59aff89b>