Lab Assignment 3 Report

Deep Learning

Rushil Ashish Shah(B20AI036)

25th February, 2023

Introduction

In this assignment, we explore the performance of ResNet18 pre-trained on ImageNet for the CIFAR100 dataset. The task is to perform classification and obtain the top-5 test accuracy for the model. We use three different optimizers: Adam, Adagrad, and RMSprop to fine-tune the pre-trained model for classification.

We plot the curves for training loss and training accuracy and analyze the results for each optimizer.

Additionally, we vary the value of attributes for each optimizer and observe its effect on the overall performance. The attributes that we vary for Adam include beta_1, beta_2, epsilon, momentum, and weight_decay. We also vary the value of lr_decay, initial_accumulator_value and weight decay for Adagrad and alpha,weight decay and momentum for RMSprop. In the report, we analyze the results of each optimizer and discuss the reasons why one optimizer performs better than the other or why not. We provide reasons for our observations and conclusions in the report. It is important to note that the use of TensorFlow is not permitted in this assignment.

Implementation, Observation and Results

Optimizer 1: Adam:

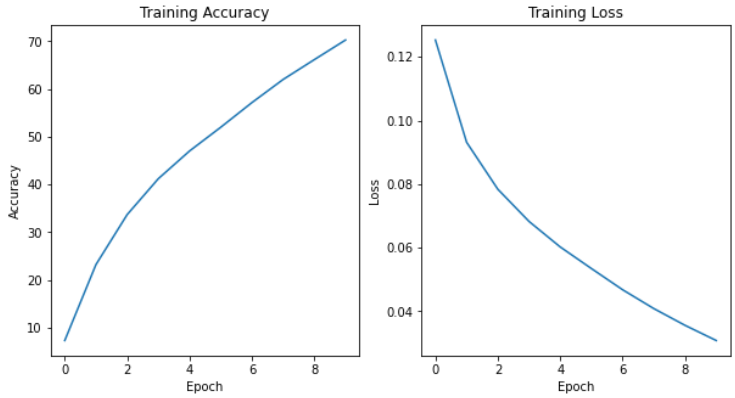
Adam is a popular optimization algorithm used in deep learning. It combines ideas from both SGD and RMSprop, making it efficient for large datasets and high-dimensional parameter spaces.

Parameters varied: Beta1, Beta2 and epsilon

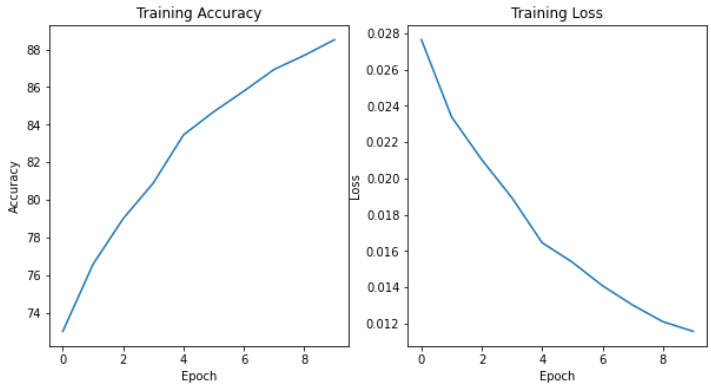
The parameters beta1 and beta2 are coefficients used in the Adam optimizer to control the exponential decay rates for the moving averages of the gradient and its square, respectively. A lower value of beta1 and beta2 would lead to a stronger weightage to recent values, while higher values would lead to a smoother estimate over time. Epsilon is a small positive constant added to the denominator of the update rule to avoid division by zero, thus ensuring numerical stability during optimization.

Changing Beta1

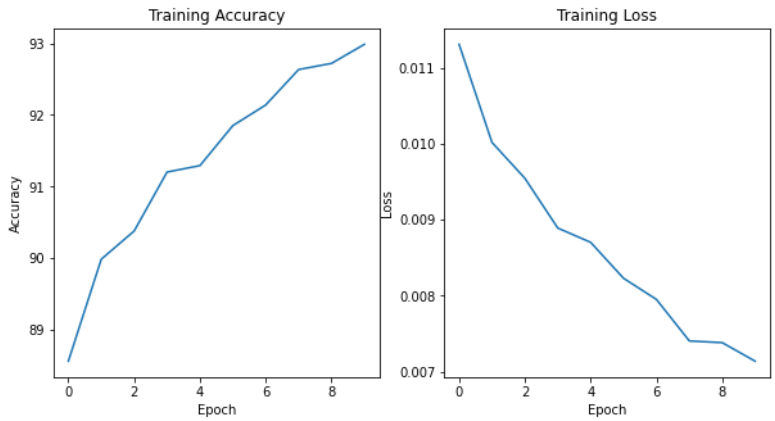
beta1 = 0.7 `Top-5 Testing Accuracy: 71.12%`



beta1 = 0.8 `Top-5 Testing Accuracy: 71.02%`



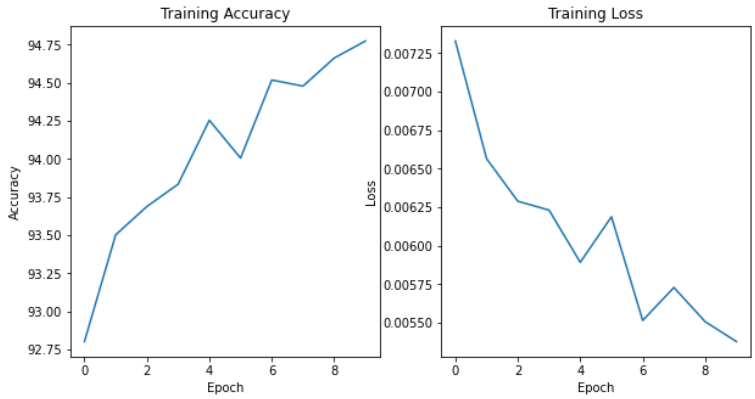
beta1 = 0.9 `Top-5 Testing Accuracy: 70.69%`



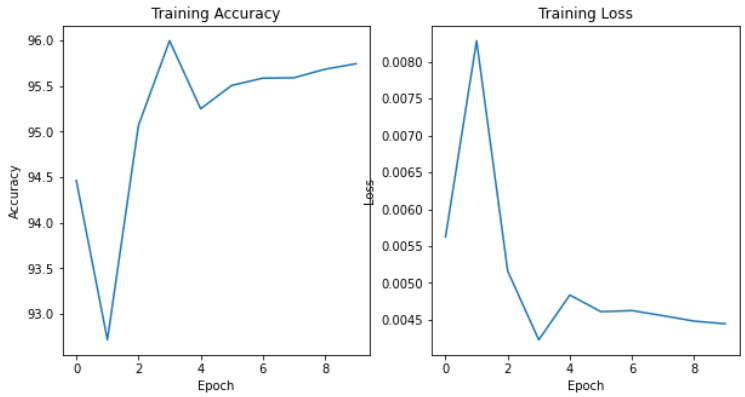
We can conclude that the parameter beta1 should set to a value close to 1 (e.g., 0.9). A higher value of beta1 can lead to faster convergence during the initial phase of training, while lower values can result in more exploration of the parameter space.

Changing Beta2

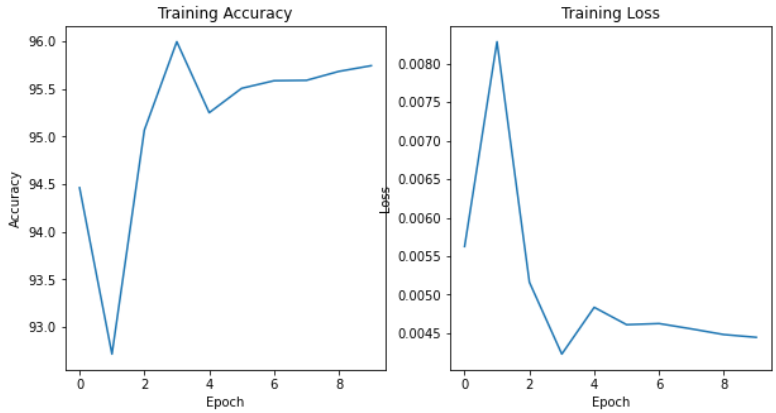
Beta2 = 0.999 `Top-5 Testing Accuracy: 70.49%`



Beta2 = 0.9995 `Top-5 Testing Accuracy: 70.48%`



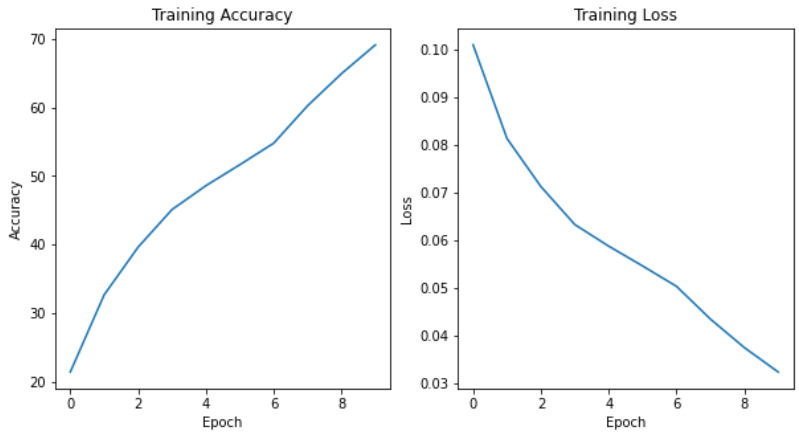
Beta2 = 0.9999 `Top-5 Testing Accuracy: 69.72%`



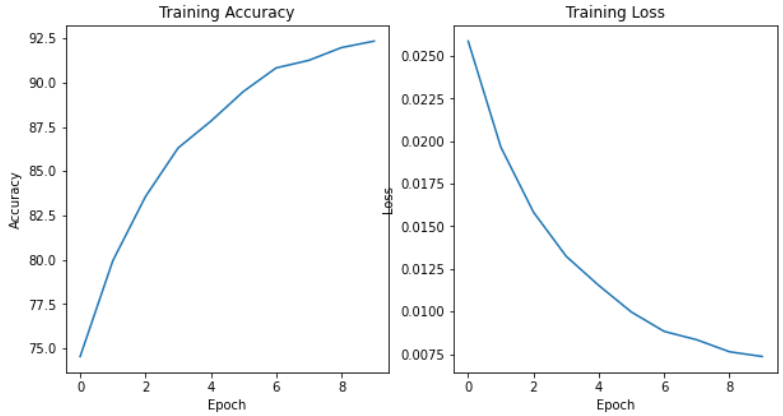
The test accuracy decreases on increasing the beta2. Although the difference is not very high it is significant enough. A small value of beta2 (e.g., 0.999) puts more weight on the past gradients and results in a smoother learning curve.

Changing Epsilon

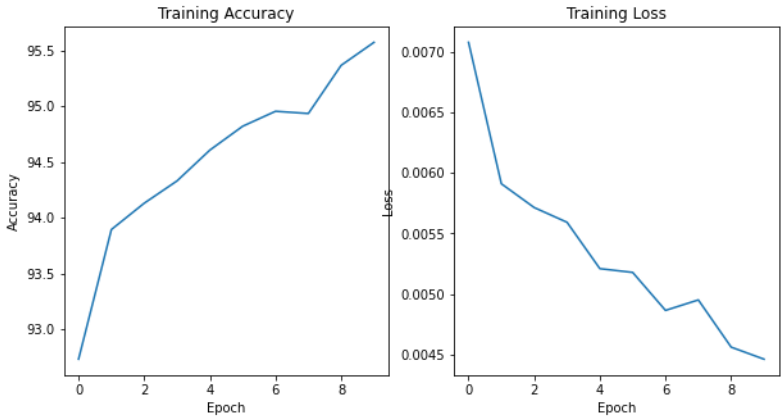
Epsilon = 1e-8 `Top-5 Testing Accuracy: 79.26%`



Epsilon = 1e-6 `Top-5 Testing Accuracy: 75.24%`



Epsilon = 1e-4 `Top-5 Testing Accuracy: 74.15%`



Accuracy decreases on increasing epsilon and Larger values of epsilon may result in more conservative learning rates that change more slowly. The default value of 1e-8 gives best result

Optimizer2: RMSprop

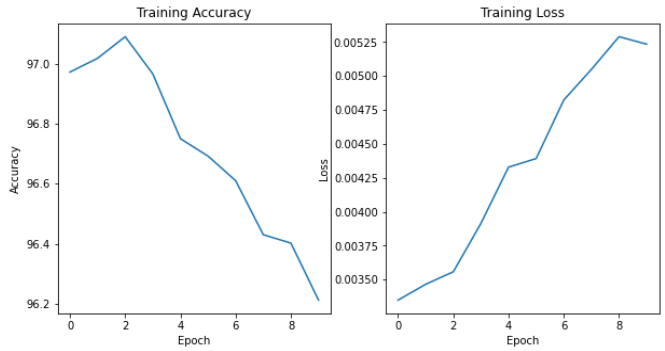
RMSprop is an optimization algorithm for neural networks that uses a moving average of the squared gradients to normalize the gradient update. It scales the learning rate based on the magnitude of the gradient, and allows for different learning rates to be used for different weights.

Parameters varied: Alpha, Momentum and weight_decay

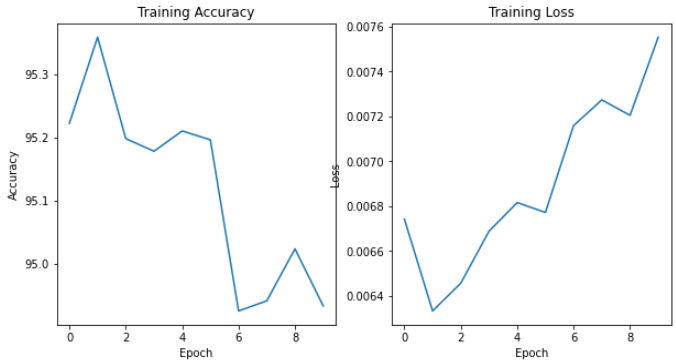
In RMSprop optimizer, the parameter alpha controls the smoothing factor for the moving average of the squared gradient. The momentum parameter controls the amount of past gradients that contribute to the current gradient update. The weight decay parameter adds a penalty term to the loss function, encouraging smaller weights and preventing overfitting.

Changing Alpha

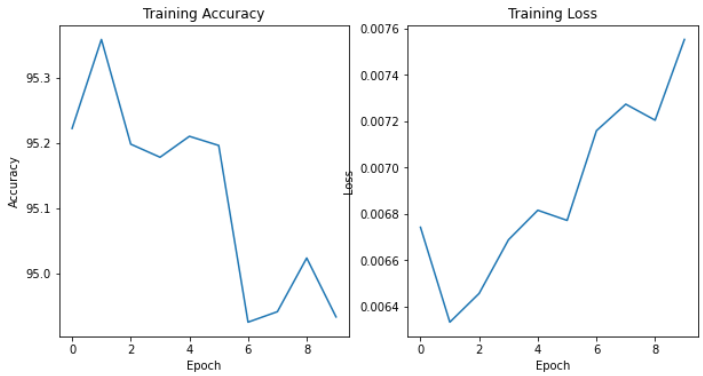
Alpha = 0.8 `Top-5 Testing Accuracy: 74.07%`



Alpha = 0.9 `Top-5 Testing Accuracy: 73.81%`



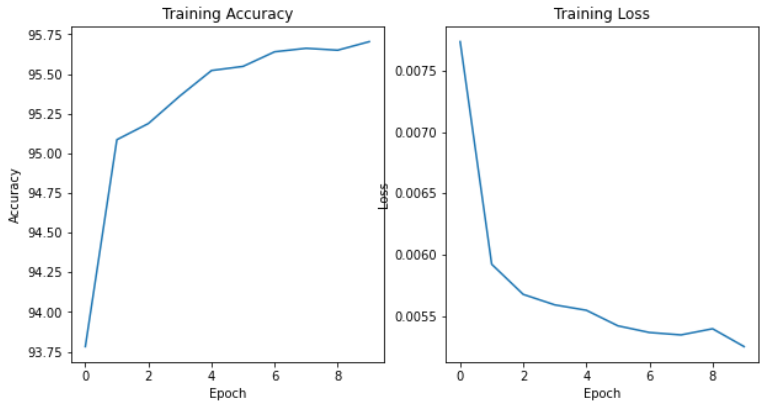
Alpha = 0.99 `Top-5 Testing Accuracy: 72.50%`



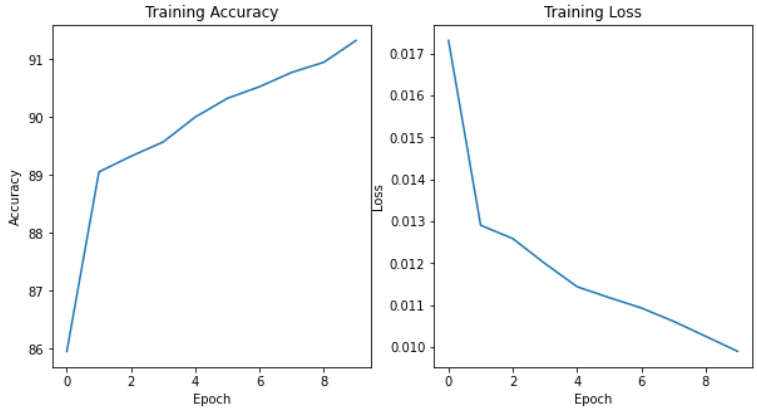
- **We can clearly see that a smaller alpha gives better test accuracy in this case.**
- **A smaller alpha value results in smaller updates to the weights at each iteration, which can lead to slower convergence but may also prevent overshooting the optimal weights.**
- **Conversely, a larger alpha value can lead to faster convergence but may result in overshooting the optimal weights and potentially diverging from the solution.**

Changing Momentum

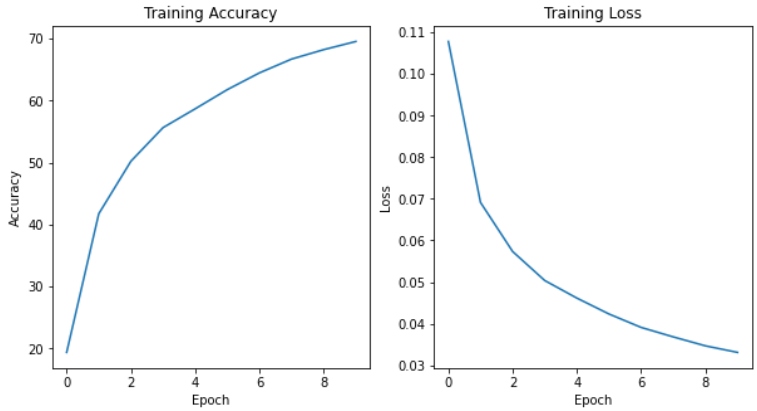
Momentum = 0.0 `Top-5 Testing Accuracy: 70.31%`



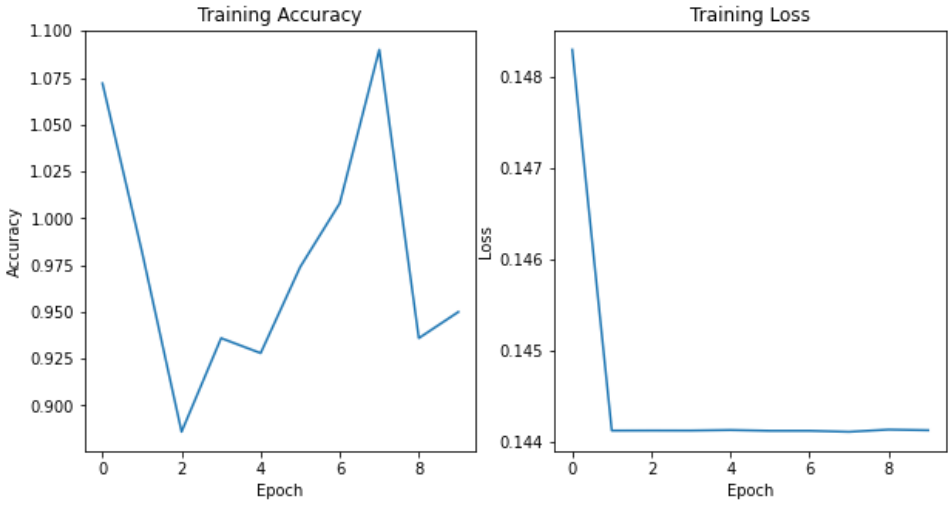
Momentum = 0.4 `Top-5 Testing Accuracy: 70.31%`



Momentum = 0.8 `Top-5 Testing Accuracy: 68.26%`



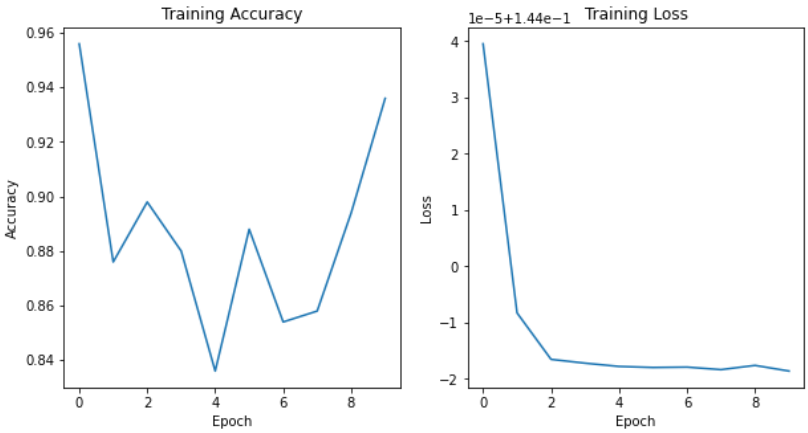
Momentum = 0.9 `Top-5 Testing Accuracy: 5%`



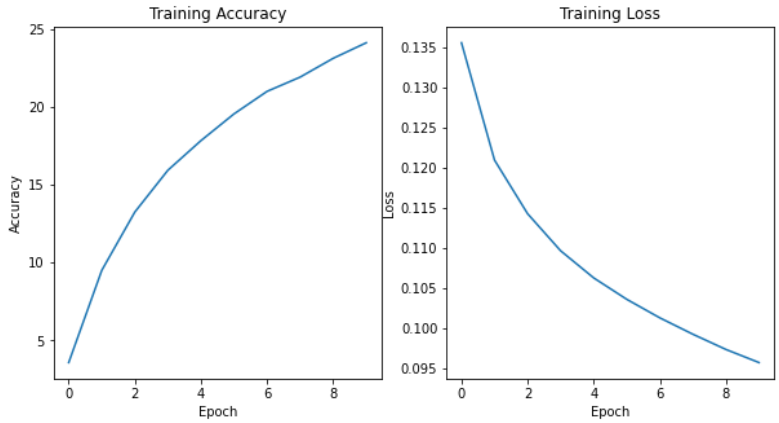
Lower momentum values are performing better because high momentum values are causing overshooting and preventing convergence.

Changing weight_decay

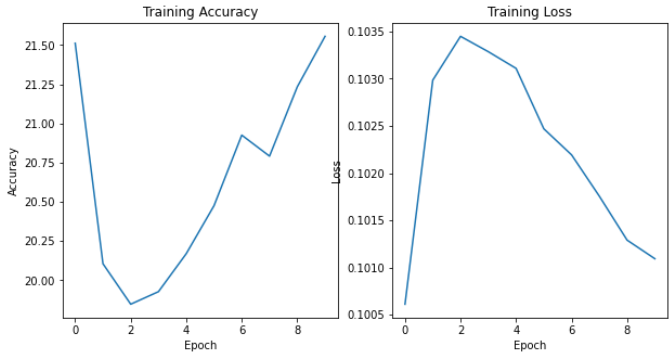
Weight_decay = 0 `Top-5 Testing Accuracy: 5.00%`



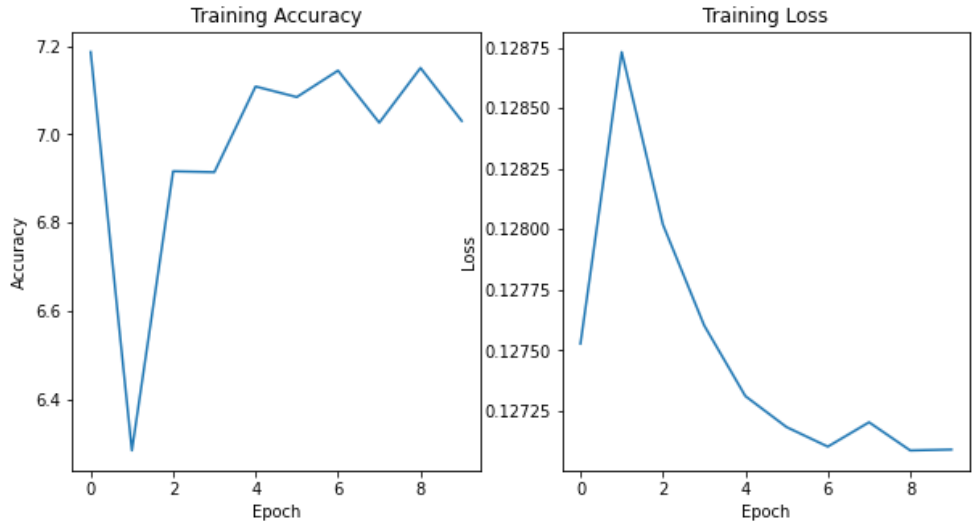
Weight_decay = 1e-3 `Top-5 Testing Accuracy: 53.24%`



Weight_decay = 1e-2 `Top-5 Testing Accuracy: 43.18%`



Weight_decay = 1e-1 `Top-5 Testing Accuracy: 23.61%`



No particular trend is visible.

A higher value of weight decay can lead to stronger regularization and better generalization performance, but if set too high, it can cause underfitting. On the other hand, setting weight decay too low can result in overfitting.

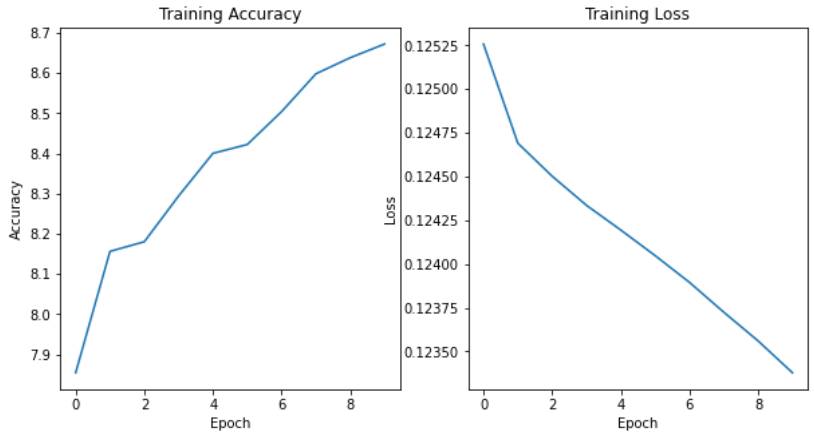
Optimizer 3: Adagrad

Adagrad is an optimization algorithm used for gradient-based optimization. It adapts the learning rate of parameters during training by scaling the learning rate inversely proportional to the root of the sum of historical squared gradients. It is well suited for sparse data & is used for NLP tasks.

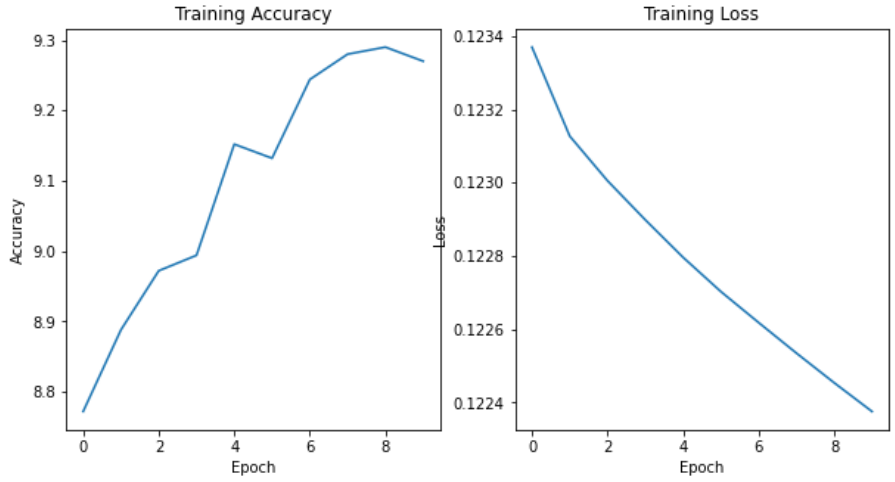
Parameters varied: weight_decay, lr_decay and intial_accumulator_value_parameter

Changing Weight_decay:

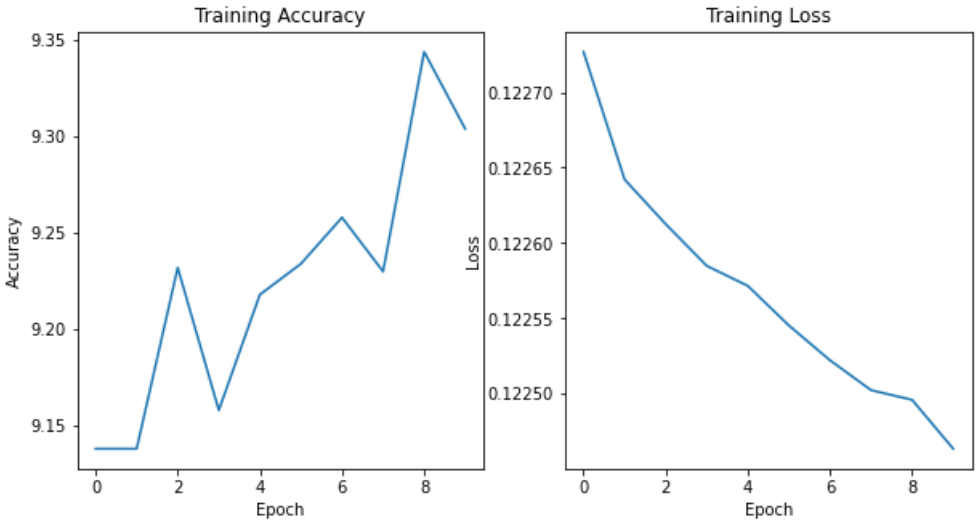
Weight_decay = 0 `Top-5 Testing Accuracy: 27.73%`



Weight_decay = 1e-3 `Top-5 Testing Accuracy: 28.82%`



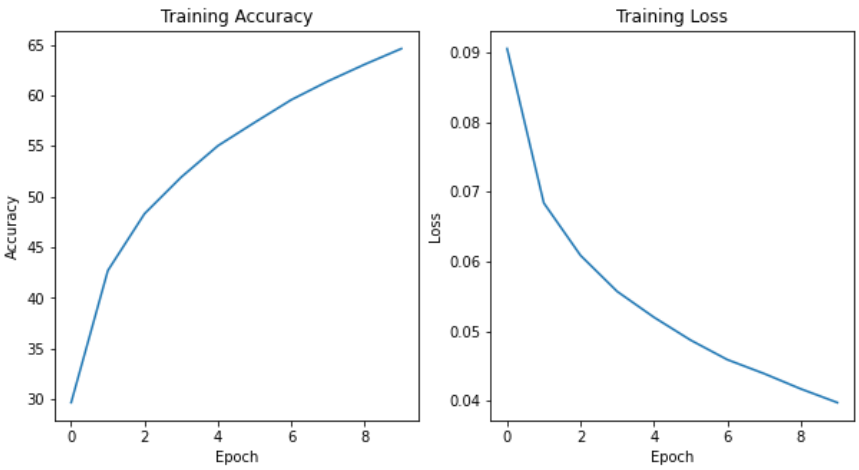
Weight_decay = 1e-2 `Top-5 Testing Accuracy: 28.42%`



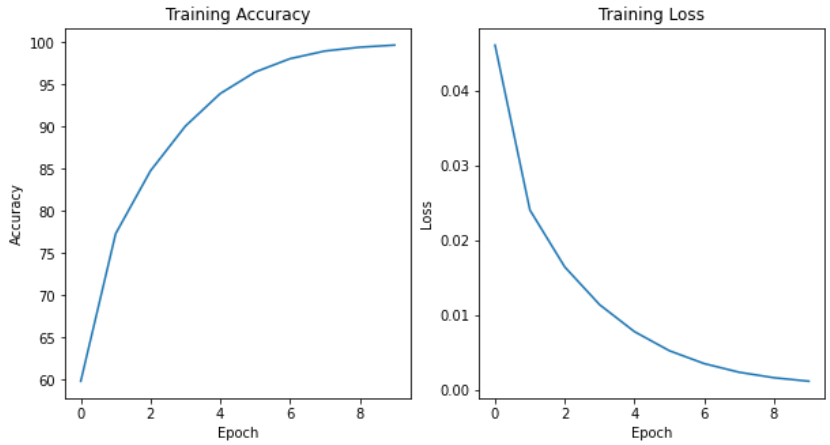
No particular trend is visible. Similar to RMSprop

Changing lr_decay:

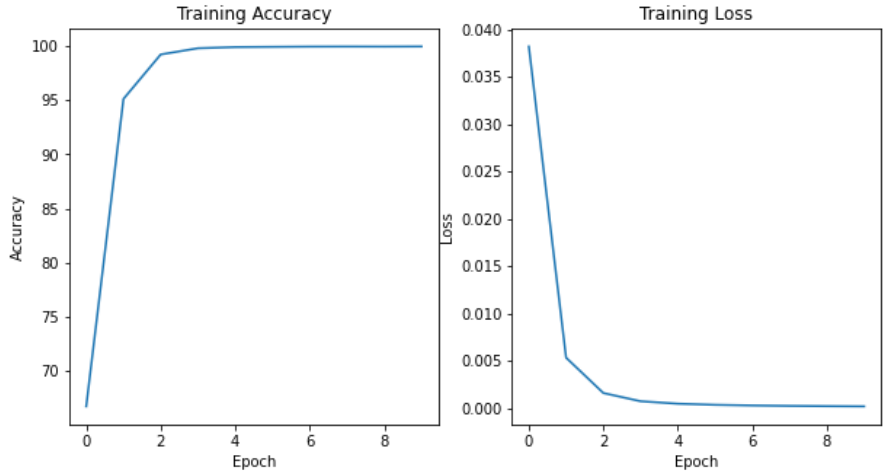
Lr_decay = 0.0 `Top-5 Testing Accuracy: 81.62%`



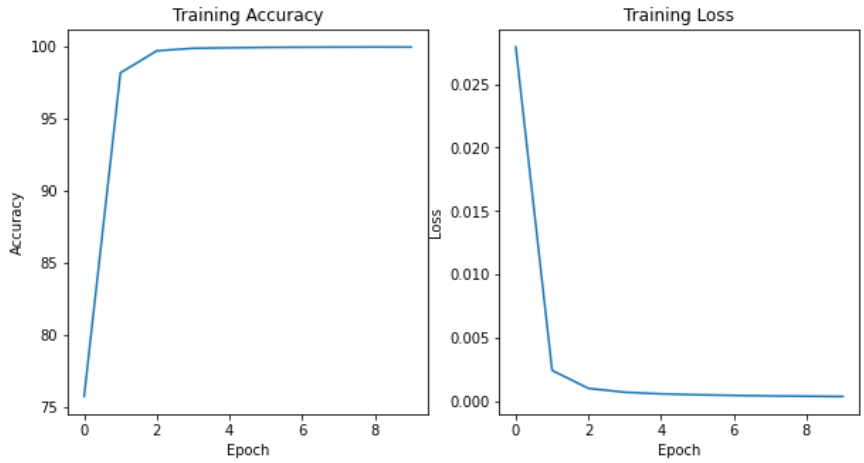
Lr_decay = 1e-5 `Top-5 Testing Accuracy: 81.34%`



Lr_decay = 1e-4 `Top-5 Testing Accuracy: 81.50%`



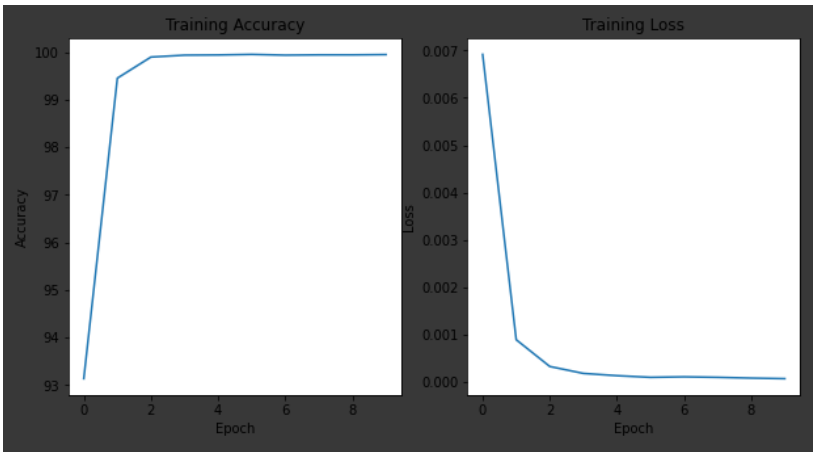
Lr_decay = 1e-3 `Top-5 Testing Accuracy: 81.56%`



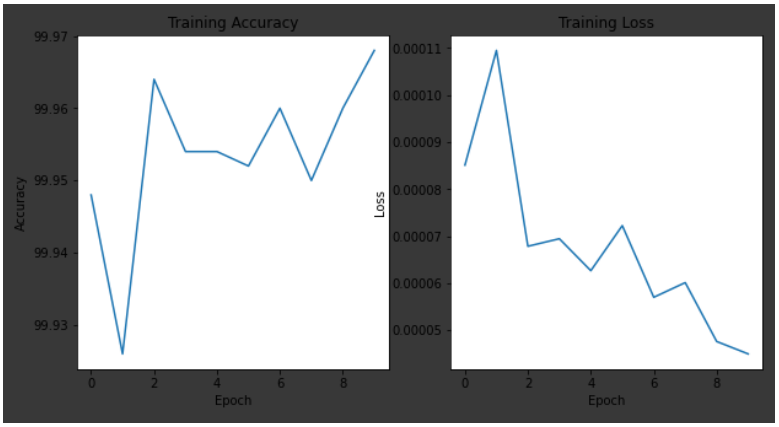
No trend observed. Hence the optimal value of lr_decay depends on the specific problem and should be chosen through experimentation.

Changing intial_accumulator_value_parameter:

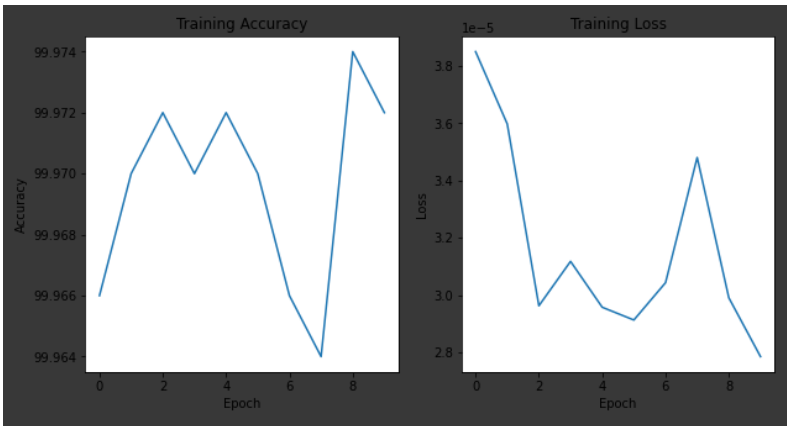
initial_accumulator_value = 0.1 `Top-5 Testing Accuracy: 81.45%`



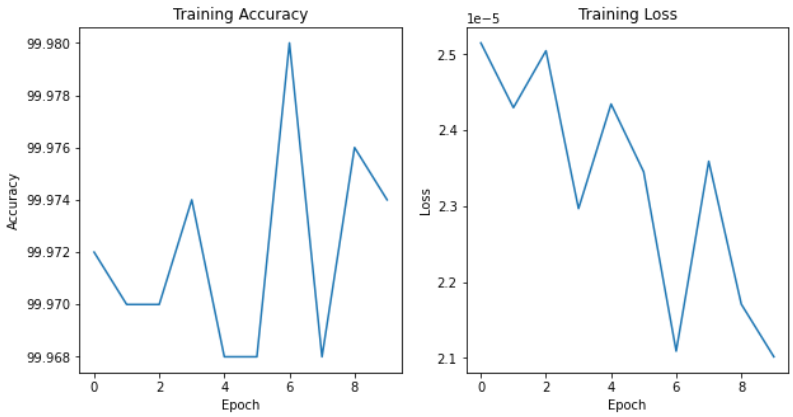
initial_accumulator_value = 0.3 `Top-5 Testing Accuracy: 81.35%`



initial_accumulator_value = 1.0 `Top-5 Testing Accuracy: 81.35%`



initial_accumulator_value = 2.0 `Top-5 Testing Accuracy: 81.45%`



No particular trend observed. The initial_accumulator_value parameter controls the starting value of the sum of squares of gradients. A higher value means the optimizer will initially give more weight to the past gradients and less to the current gradients. This can result in slower convergence but can be useful for highly non-stationary objectives. On the other hand, a lower value can result in faster convergence but can be unstable for highly non-stationary objectives.

Conclusion

- Adagrad performs the best on CIFAR 100 dataset giving maximum top5 test accuracy of 81.45%. This generally happens when the variance of gradients is high.
- In the preprocessing part it was observed that the pretrained model was overfitting the data and hence I used some transformation to prevent the same.
- Beta1 in Adam should be as close to 1 for faster convergence during the initial phase of training
- Beta 2 in Adam should be small as it puts more weight on the past gradients and results in a smoother learning curve.
- Epsilon in Adam should be small. The default value 1e-8 gives the best accuracy.
- A smaller alpha value for RMSprop gives better test accuracy
- Lower momentum values for RMSprop are performing better because high momentum values are causing overshooting and preventing convergence.