Assignment-5 Training Arc

Jaidev Shriram (2018101012)

April 18, 2021

The primary model, which is modified throughout, consists of four convolutions with a 4x4 kernel, padding 2m and single stride. This is followed by a linear layer with 256 input features and 61 output features. The image sized used is 64×64. There is a ReLU activation function between each layer.

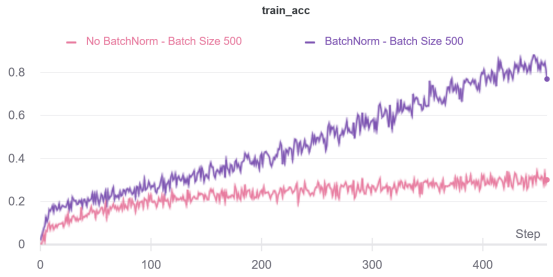
1 Impact of Dropout

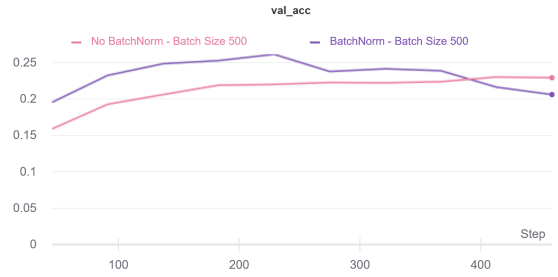
Here, we train the model with and without dropout. As expected, the model with dropout is able to generalize better - as seen with a higher validation accuracy over various epochs. This is because dropout effectively acts as a regularizer, preventing the network from easily overfitting on training data.

2 Choice of Activation Function

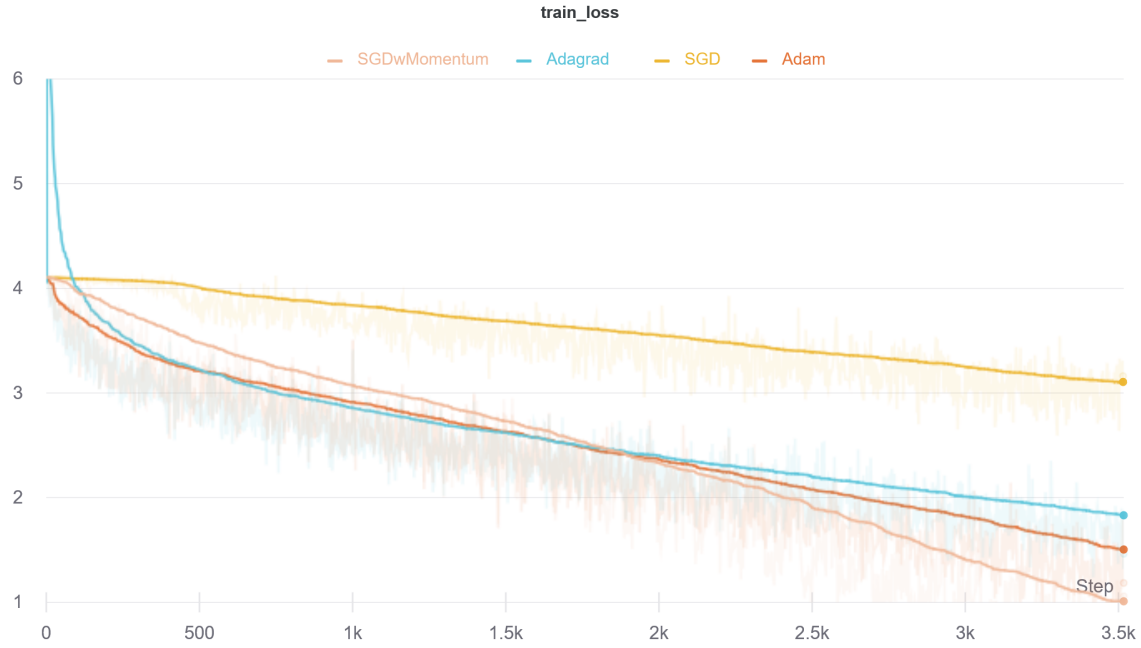
Here we compare the use of three non-linear activation functions - Sigmoid, ReLU and LeakyReLU. ReLU gives the best result with a consistently better accuracy for train and validation sets. This is because the gradient flow is a lot better in ReLU as the output of ReLU is defined as max(0, in). Hence, we can get bigger gradients with ReLU, leading to a better model. Tanh performs better than Sigmoid since the derivates of Tanh are also bigger than Sigmoid. Note that a larger gradient will likely result in faster convergence since we are taking bigger steps.

3 With BatchNorm





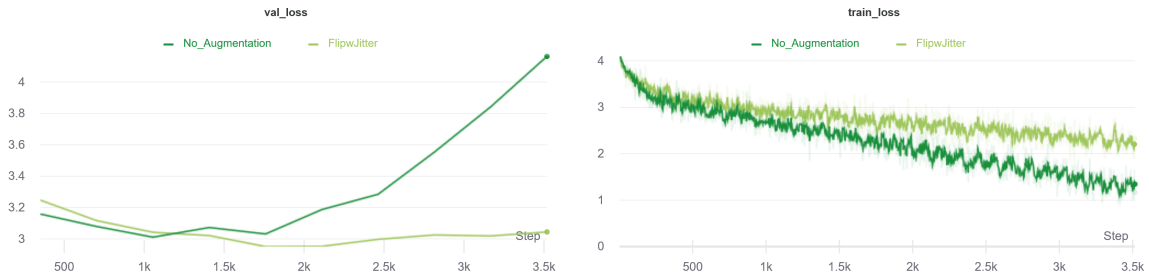
Batch Normalization is a technique to standardize the distribution of the inputs to a layer for a mini-batch. It typically stabilizes the learning process and also provides some regularization. We notice this experimentally as well: Our model is able to get a higher training accuracy with batchnorm while simultaneously generalizing better on the validation set.



4 Different Optimizers

We try four different optimizers - SGD, SGD with momentum, Adagrad, and Adam. SGD with momentum is noticeably better than simple SGD as the momentum term helps avoid local minimas. The other optimizers also have a notion of momentum, leading to similar results. Performance of adagrad detererotates over time likely because the sum of gradient square can get really big, resulting in smaller updates. Adam does well since it combines the idea behind Adagrad and SGD with momentum.

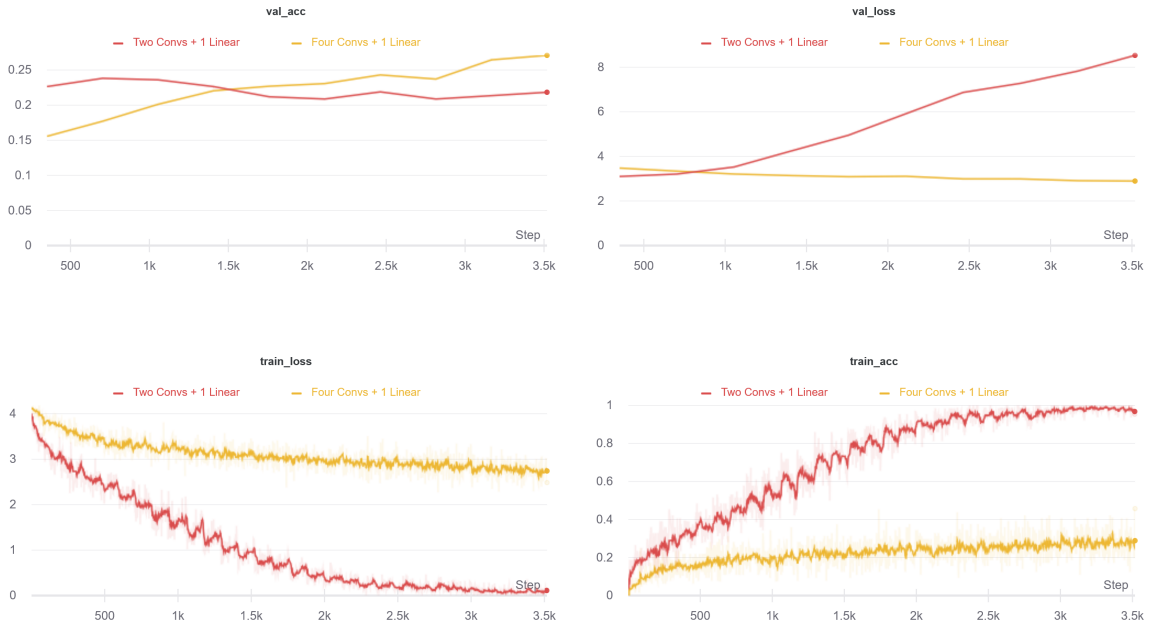
5 Effect of Augmentation



Here we have two models - one without any augmentation and another with two augmentations "Random HorizontalFlip and Random Colour Jittering - Hue, Brightness, Saturation, Contrast". Notice how the training loss decreases quicker withoout any augmentation. On the contrary, the

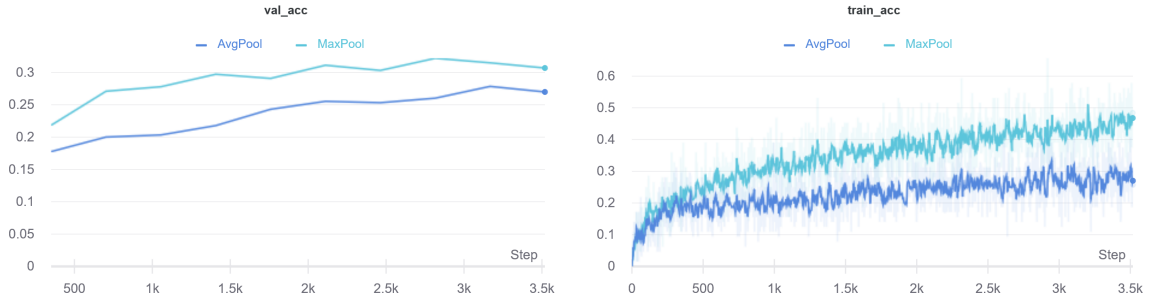
validation loss increases over time - indicating that the model is overfitting. Clearly, the model with augmentation doesn't suffer from this as the validation loss doesn't change much.

6 Number of Layers



Here, we have two models - one with four convolution/ReLU/MaxPool blocks followed by a Linear layer and another with two such blocks and a Linear layer. Since the first model downsamples the image to a greater extent, the total number of parameters are actually fewer compared to the model with just two Conv layers. Models with more parameters are prone to overfitting - which is evident here.

7 Choice of Pooling



AvgPool gives poorer results on both training and validation sets compared to max pool. This is likely because the smoothening the image (by taking an average of an area) removes essential features while max pool is able to select the most distinctive pixels in a window.

8 Main Report

The main report can be viewed here: https://wandb.ai/jaidev/assignment-5-jaidevshriram/reports/Assignment-5–Vmlldzo2MTg4ODE?accessToken=ltodwnxrez4nbvjfj5jgpr1arzwz86j146hw671taju3u7p966c7vjg9yxnh19bj

References

[1] Weights and biases.