CS579 – Homework 1 Solutions

Opeyemi Ajibuwa

Task 1

a. My choice of framework was PyTorch. The hyperparameters that led to the best performance on each of the models were used to train the models and are summarized as follows:

Model	Training configurations/Hyperparameters
LENET on MNIST	**learning rate**: 0.3; **batch size**: 64; **optimizer**: SGD; **loss function**: CrossEntropyLoss; **epochs**: 40
LENET on CIFAR-10	**learning rate**: 0.001; **batch size**: 32; **optimizer**: Adam; **loss function**: CrossEntropyLoss; **epochs**: 40
ResNet18 on MNIST	**learning rate**: 0.01; **batch size**: 128; **optimizer**: Adam; **loss function**: CrossEntropyLoss; **epochs**: 40;
ResNet18 on CIFAR-10	**learning rate**: 0.01; **batch size**: 32; **optimizer**: SGD; **loss function**: CrossEntropyLoss; **epochs**: 40; **momentum**: 0.9
VGG16 on MNIST	**learning rate**: 0.001; **batch size**: 64; **optimizer**: SGD; **loss function**: CrossEntropyLoss; **epochs**: 40; **momentum**: 0.9
VGG16 on CIFAR-10	**learning rate**: 0.01; **batch size**: 128; **optimizer**: SGD; **loss function**: CrossEntropyLoss; **epochs**: 40

b. The plots of the accuracy and loss of the 6 models are shown in the following figures:

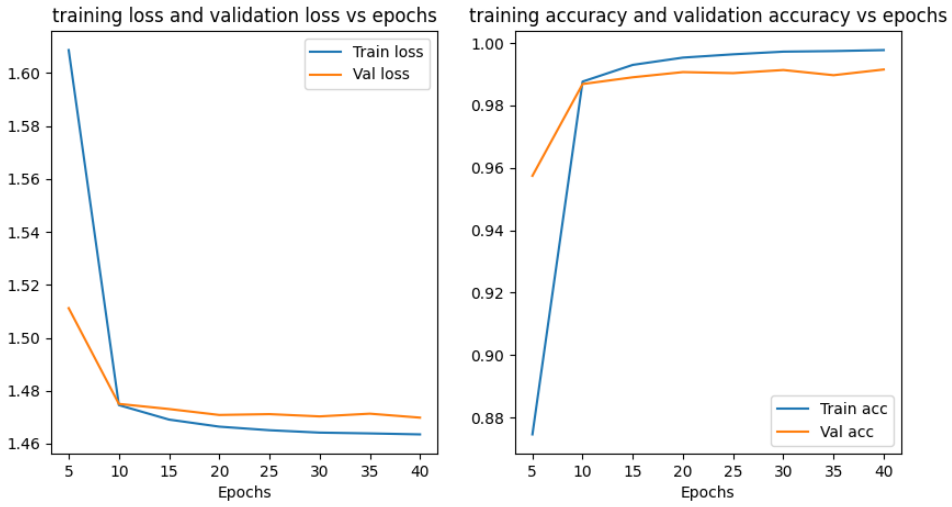

Fig. 1a: LeNet with MNIST

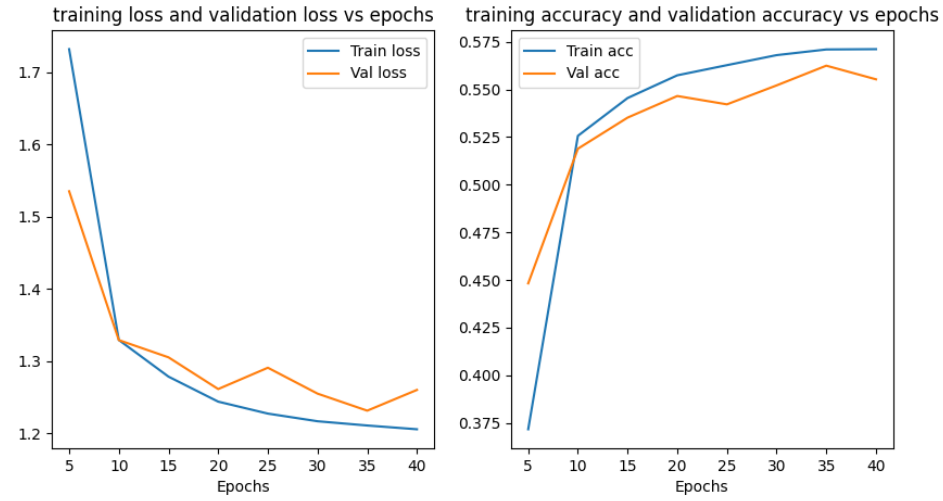

Fig. 1b: LeNet with CIFAR-10

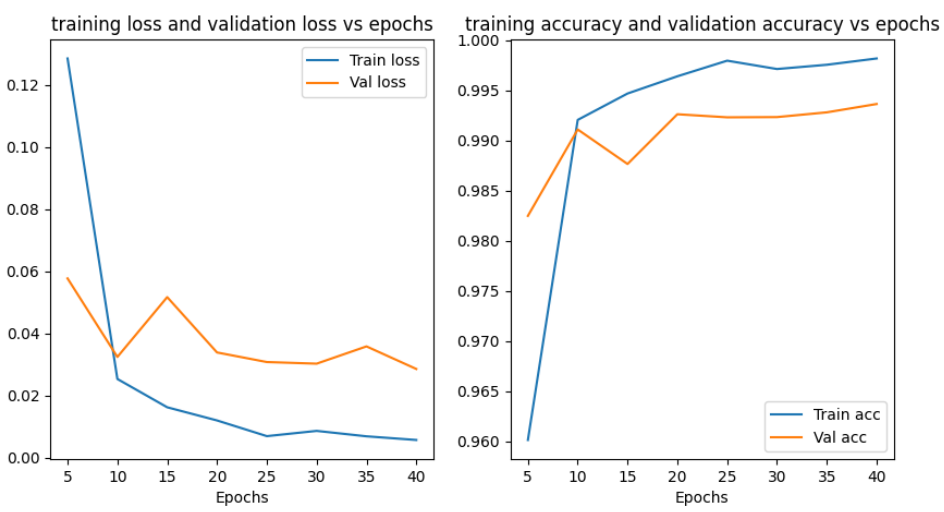

Fig. 1c: ResNet18 with MNIST

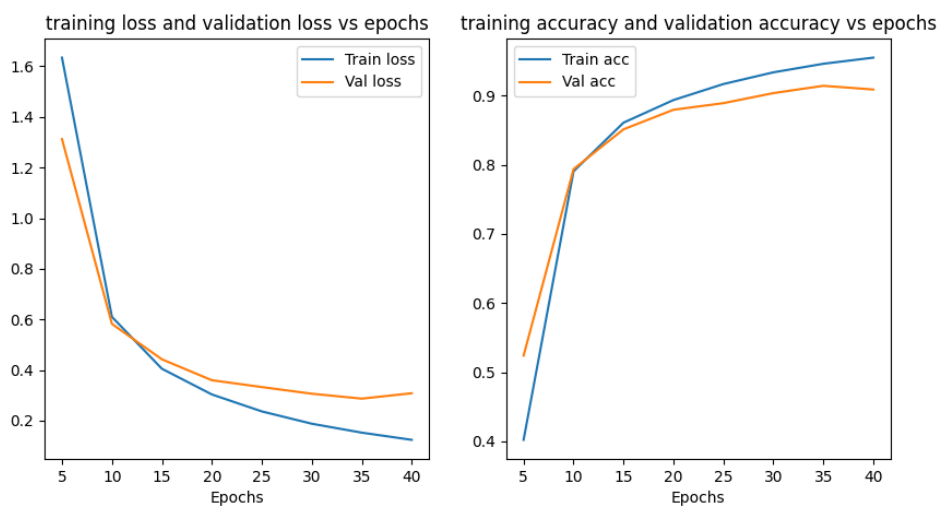

Fig. 1d: ResNet18 with CIFAR-10

Fig. 1e: VGG16 with MNIST

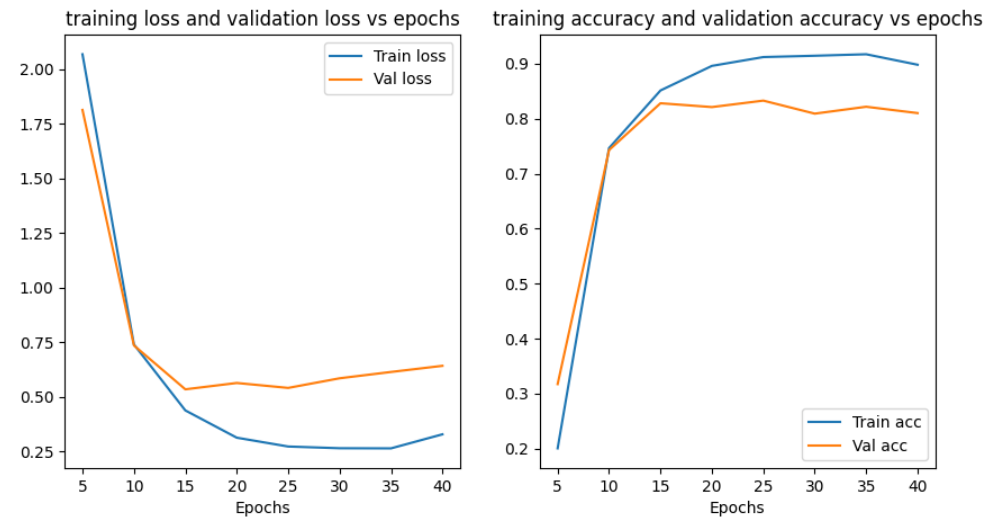

Fig. 1f: VGG16 with CIFAR-10

c. The loss and accuracy of the six models all behaved quite differently on the training and validation datasets. In the LeNet with MNIST model, both training and validation loss decreased linearly until the 10th epoch and therefore maintained a gradual descent. Similarly, the training and validation accuracy increased with the number of iterations converging towards 99% at the tail end of the training. In contrast, the LeNet model achieved an average performance on the CIFAR-10 dataset. This could be as the result of the simple architecture of the LeNet and the corresponding large size/structure of the CIFAR-10 dataset suggesting it may not be the best choice.

The ResNet model seems to overfit on the MNIST dataset as seen in Fig 1(c). There was a big drop-off in the accuracy and a considerable difference in the training and validation loss in the first few epochs suggesting that the model overfitted. On the other hand, the LeNet with CIFAR-10 model's performance was smooth over all epochs in both training and validation. This can be explained by the deep architecture of ResNet being sufficiently optimized for the CIFAR-10 dataset.

The VGG16 model on the MNIST dataset appeared excessively complex and overparameterized for such simple data like the MNIST dataset. As seen from Fig 1(e), the validation accuracy and loss had the better performance in the early iterations and increased (or decreased in terms of loss) matching the training performance in the later iterations. Similarly, VGG16 on CIFAR10 achieved a smooth performance in terms of accuracy and loss over both the training and validation sets.

Task 2

a. The plots for the LeNet18 on MNIST model and accompanying explanations are presented as follows:

I. **With rotations VS without rotations:** As seen in the figure below, an observable difference is that the validation accuracy in the LeNet18 on MNIST model was higher (and correspondingly lower in loss) and closer to the training results when the images are rotated compared to when they are not.

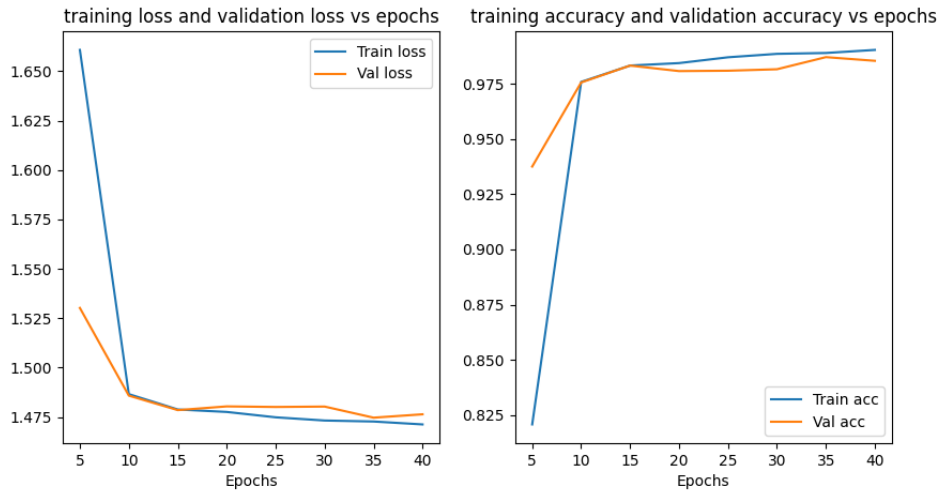

Fig. 2a: With rotations

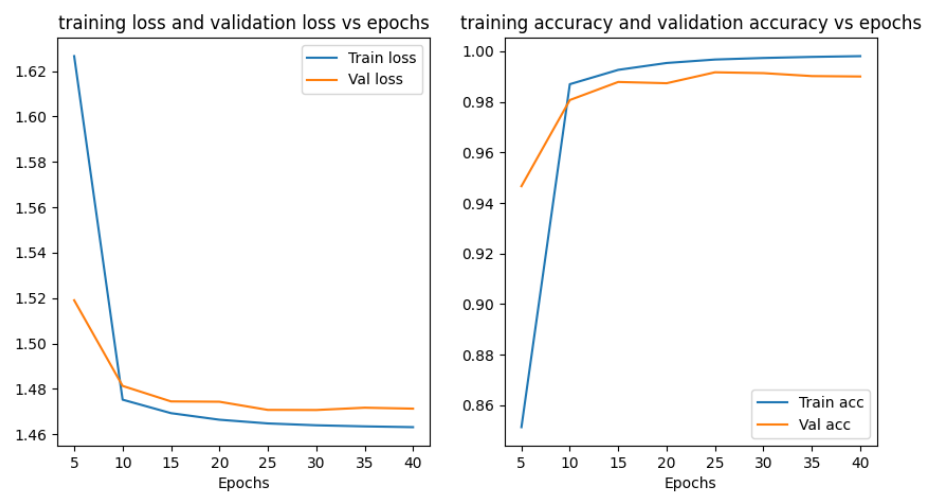

Fig. 2b: Without rotations

II. **With random horizontal flips VS without random horizontal flips:** The model achieved a higher validation accuracy and correspondingly lower loss when flipped compared to when it is not flipped as can be seen in the Figs. 2(c) and 2(d).

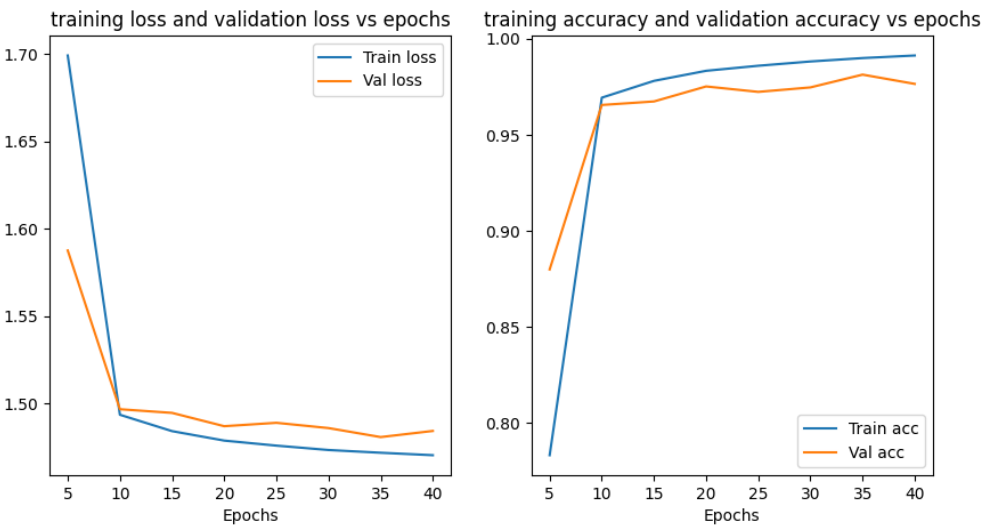

Fig. 2c: With flips

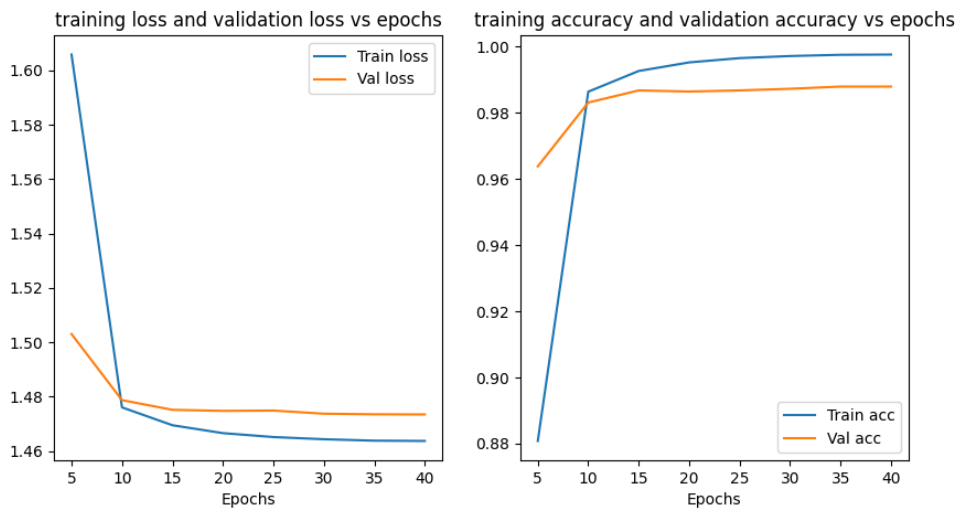

Fig. 2d: Without flips

III. **With SGD VS Adam:** The model seemed to achieve better performance with SGD optimizer than Adam. Both the training and validation loss of the model with Adam optimizer is higher than with SGD. However, please note the inversion in Fig. 2(e).

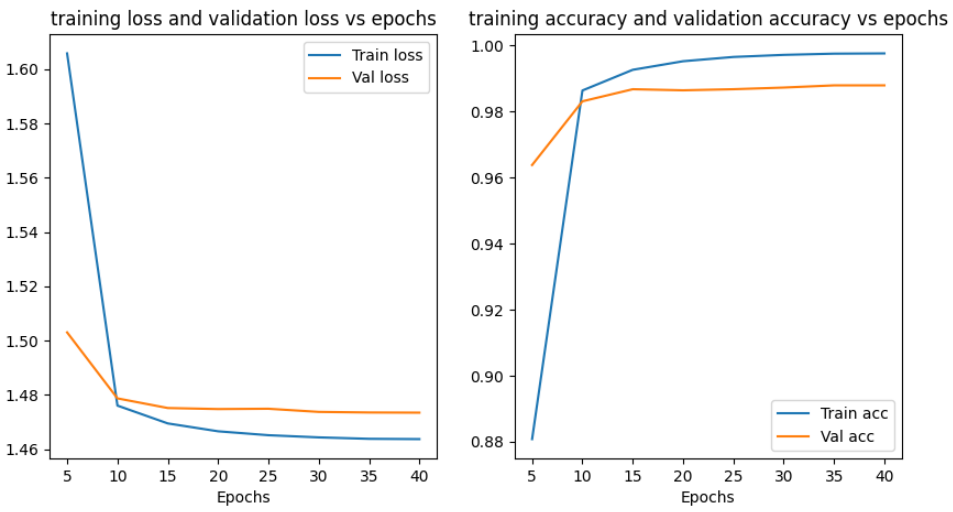

Fig. 2e: With SGD

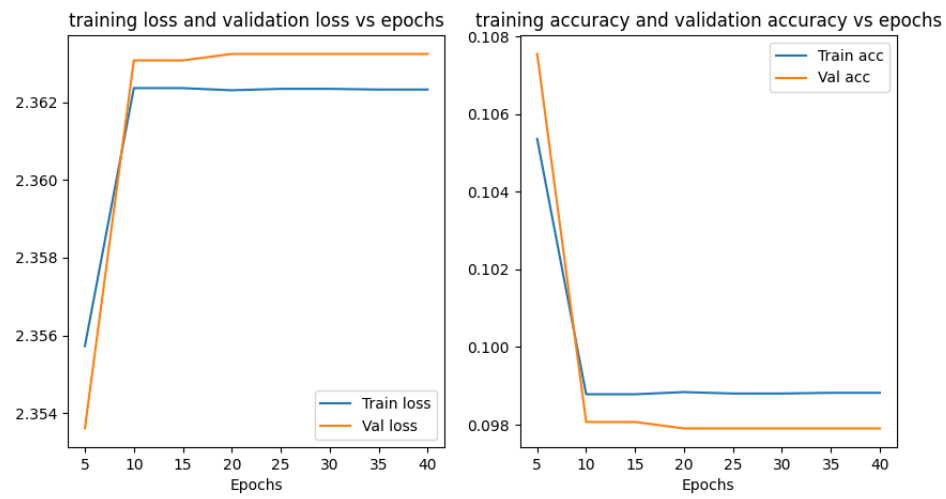

Fig. 2f: With Adam

IV. **With learning rate 0.3 VS learning rate 0.01:** As seen in the figure below. The training and validation performance obtained very similar results in both accuracy and loss when the learning rate = 0.01 is used for our model. This suggests that the model neither underfits or overfits to the training data as compared to using learning = 0.3 in which the validation accuracy decreased with increasing training epochs.

Fig. 2g: With lr = 0.3

Fig. 2h: With lr = 0.01

V. **With batch size 32 VS batch size 64:** There are no significant differences observed between using batch size 32 and batch size 64 for this model.

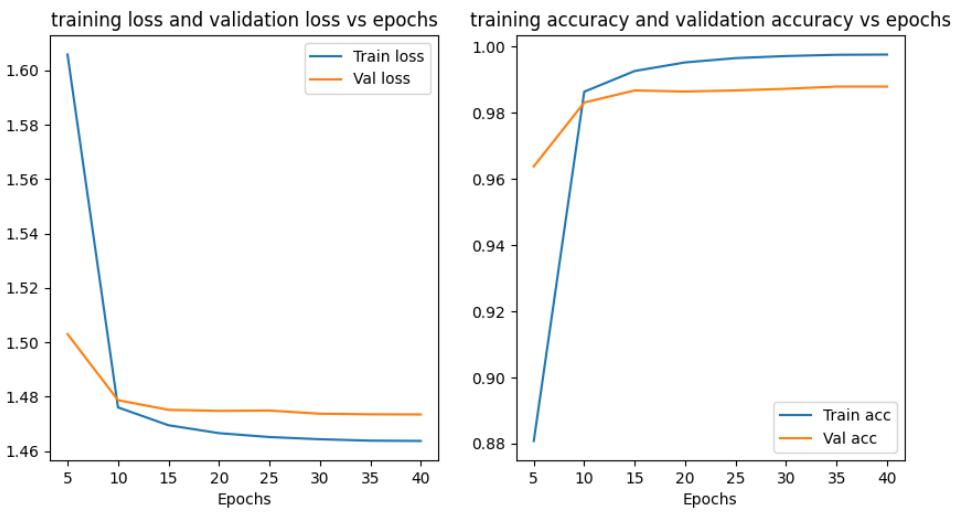

Fig. 2i: With batch size = 32

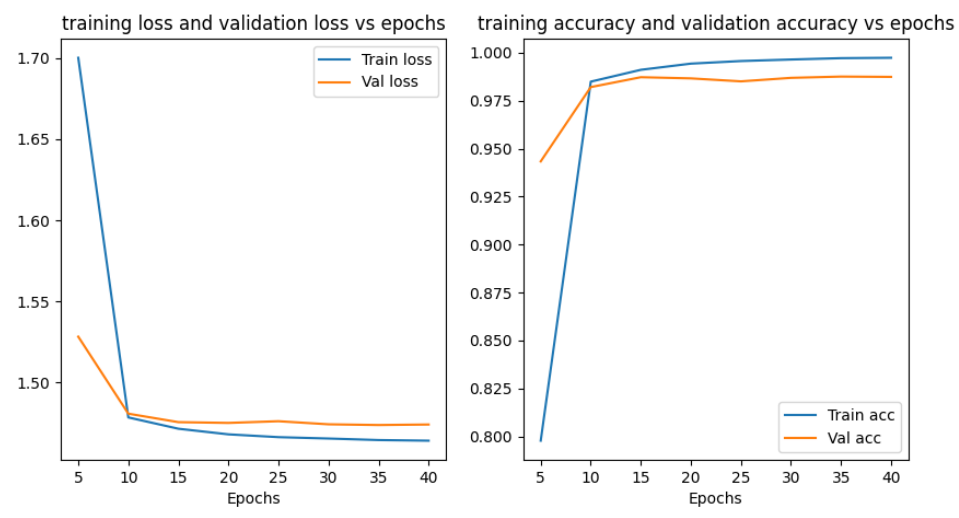

Fig. 2j: With batch size = 64

b. The plots for the ResNet18 on CIFAR-10 model and accompanying explanations are presented as follows:

I. **With rotations VS without rotations:** The validation performance seems to be better in the rotated scenario of the model. That is, the loss decreases faster towards the end of the training epochs (and correspondingly the validation accuracy keeps increasing). As seen in Fig. 3(a), the validation accuracy of the non-rotated scenario seem to flatten out towards the end of the training epoch.

Fig. 3a: With rotations

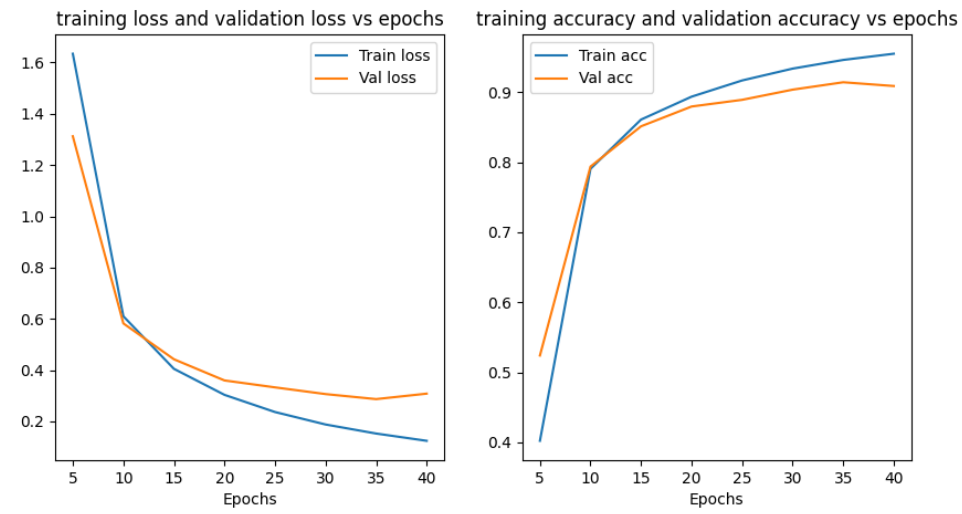

Fig. 3b: Without rotations

II. With random horizontal flips VS without random horizontal flips: The validation loss and accuracy of the model is better in the flipped scenario compared to the un-flipped model scenario. The loss decreases faster and accuracy increases faster with training epochs in the flipped model.

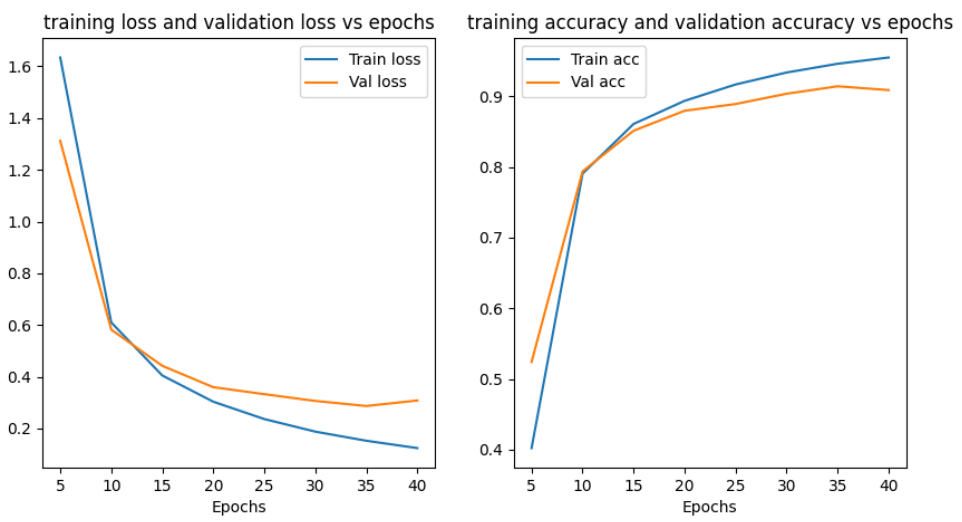

Fig. 3c: With flips

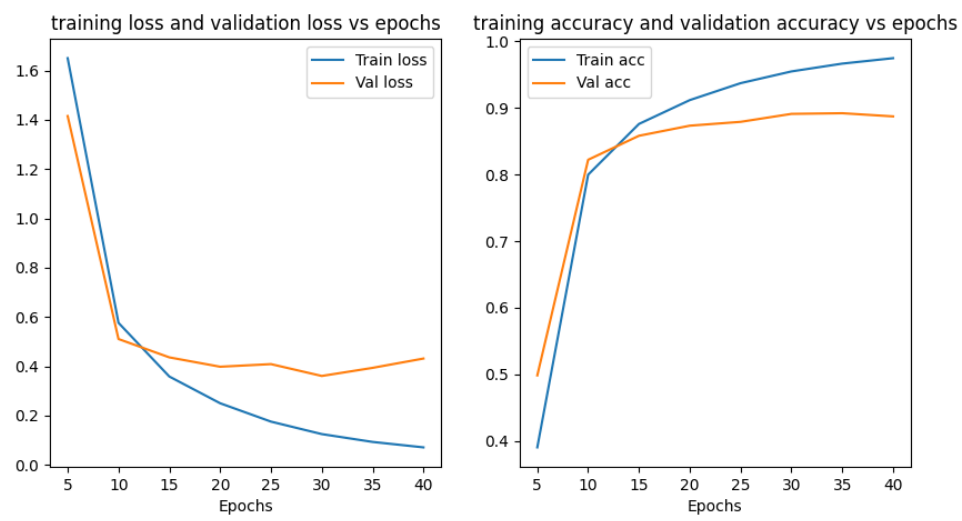

Fig. 3d: Without flips

III. With SGD VS Adam: It appears that there is no noticeable difference between the two model scenarios as seen from the figures below.

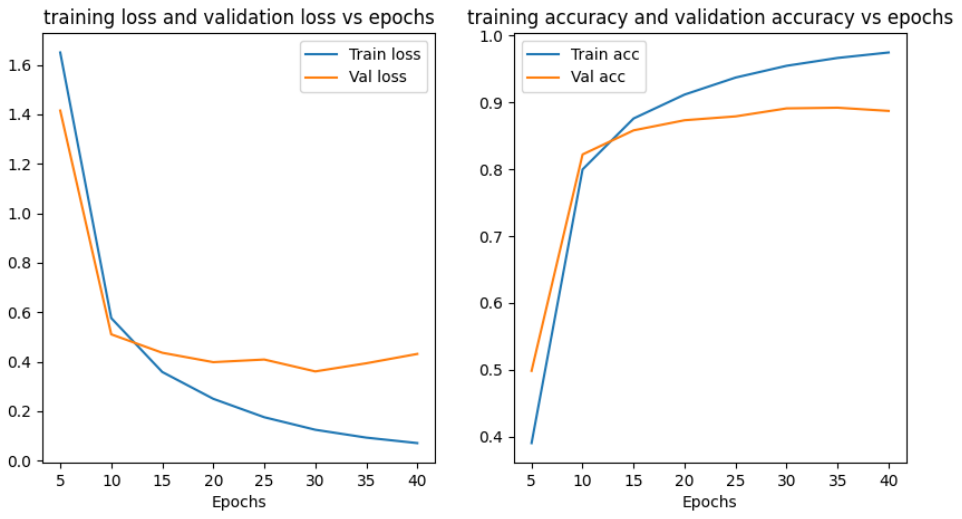

Fig. 3e: With SGD

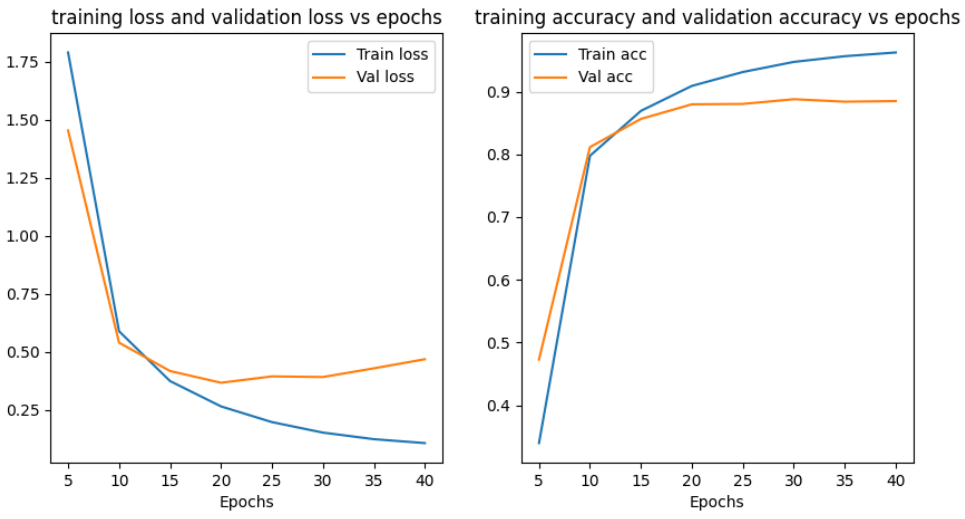

Fig. 3f: With Adam

IV. With learning rate 0.1 VS learning rate 0.01: The two model scenarios seem to achieve the same validation accuracy and lower loss with learning as epoch increases. However, the model with learning rate 0.01 seems to converge quicker than with learning rate 0.1.

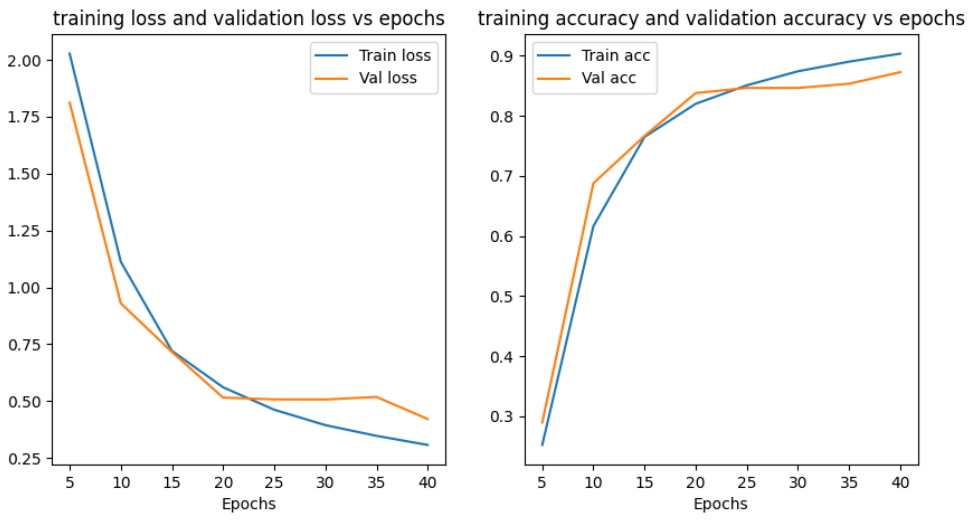

Fig. 3g: With lr = 0.1

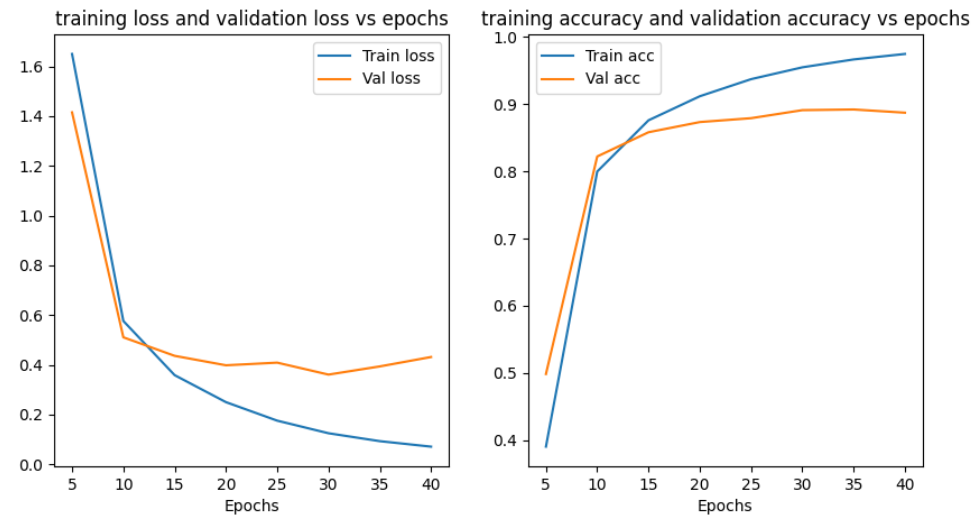

Fig. 3h: With lr = 0.01

V. **With batch size 32 VS batch size 64:** The model with batch size 32 appears smoother in its performance in terms of accuracy and loss compared to the model with batch size 64 which oscillates slightly at the later parts of the training epochs.

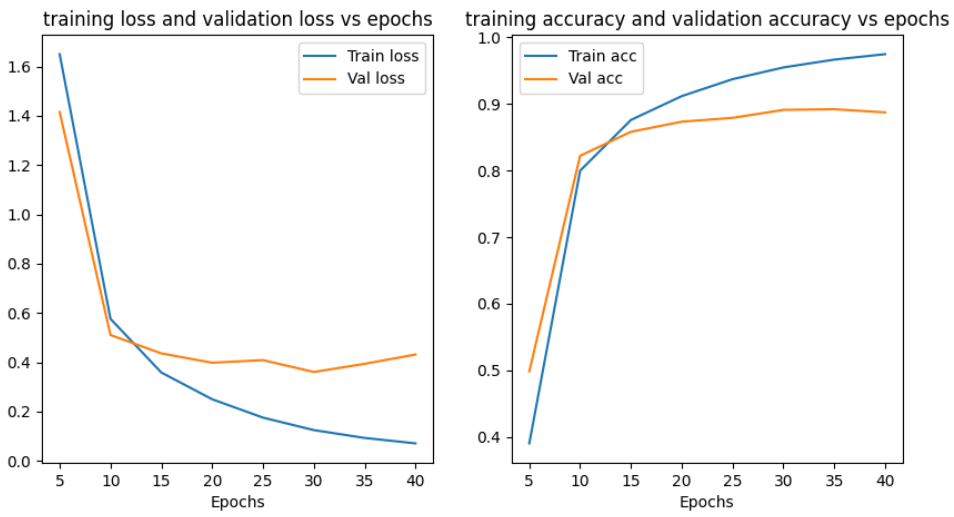

Fig. 3i: With batch size = 32

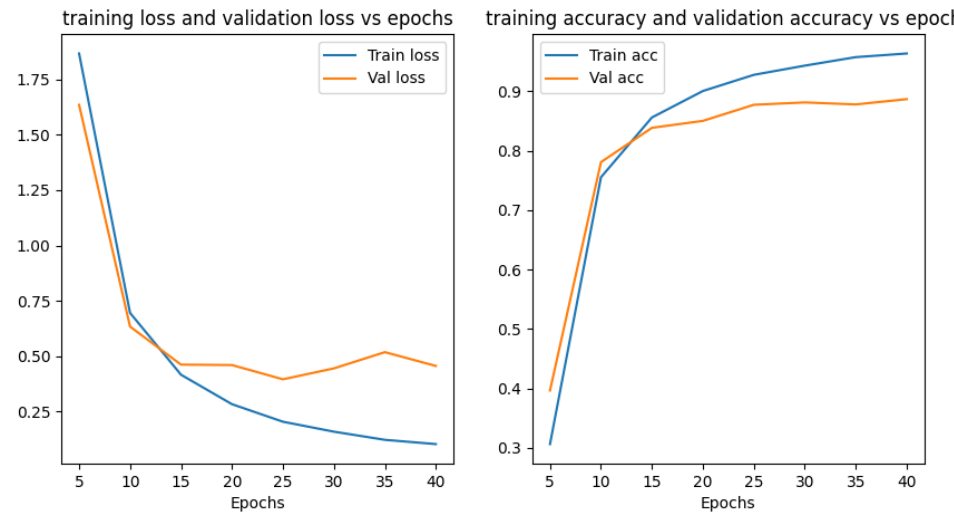

Fig. 3j: With batch size =64