ECGR 5106 – Real Time Machine Learning (Spring 2023)

Nahush D. Tambe – 801060297

https://github.com/Ntambe25

Homework # 3

Problem 1:

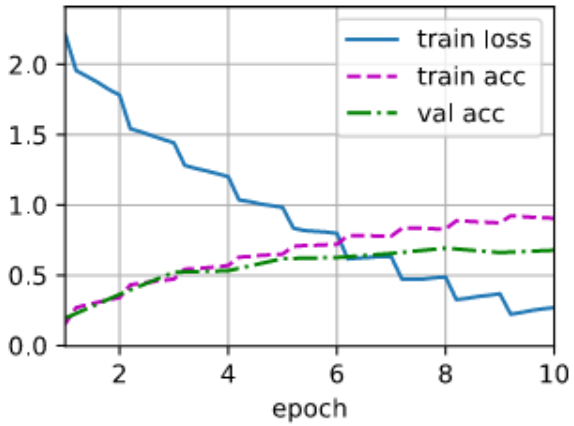

Figure 1: Results for Baseline VGG Model

Figure 1 above shows the plot which includes training loss, training accuracy, and validation accuracy for the baseline VGG model. For this model, all the images were resized to 64x64 resolution before feeding it into the model. It can be seen that the training loss starts from around 2.25 and shows a decreasing trend, going down to about 0.25. Both the training and validation accuracy start very low at around 0.25 and go up to 0.75 - 0.8. At start, the model does not show any overfitting, but as the model progresses further into training, the generalization gap increases significantly starting at around 4 Epochs.

```
Computational complexity:       637.99 MMac
Number of parameters:           34.44 M
```

Figure 2: Model Complexity for baseline VGG Model

Figure 2 above shows the model complexity and total number of parameters count for the baseline VGG model.

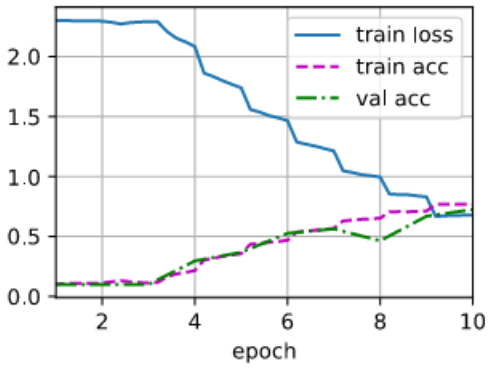

Figure 3: Results for VGG-16 Model

After plotting the baseline VGG model, the same model structure was used and few more layers were added to create the VGG-16 model with a total of 16 layers. Figure 3 above shows the results for same which include training loss, training accuracy, and validation accuracy. It can be clearly seen from the plot that the training loss starts out as almost straight line and after 3 Epochs, it starts decreasing, going down to about 0.75. On the other hand, the training and validation accuracies start very low and go up to about 0.75.

```
Computational complexity:       1.28 GMac
Number of parameters:           39.93 M
```

Figure 4: Model Complexity for VGG-16 Model

Figure 4 above shows the model complexity and total number of parameters count for the VGG-16 model.

This model, when compared to the baseline VGG model shows some improvement when the generalization is considered. But the overall values are better in the baseline model, like the training loss goes well below 0.5, but in this plot, they are around 0.75. The training accuracy was far better in the baseline model than it is in this model's plot. The validation accuracy is around the same for both the models.

In terms of the model complexity and parameters count, as the number of fully connected layers increased, so did the model complexity and the total number of parameters count.

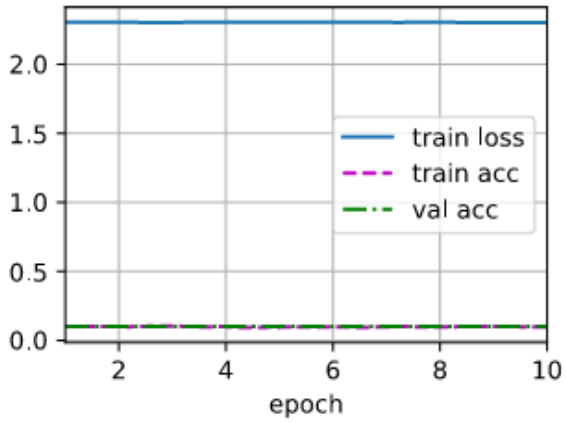

Figure 5: Results for VGG-19 Model

Using the same structure as VGG-16 model, only adding 3 more Conv2d layers, this model was created, which is commonly known as VGG-19 model. Figure 5 above shows the results for same which include training loss, training accuracy, and validation accuracy. It can be clearly seen that the plots shows straight, zero slope lines for training loss, training accuracy and validation accuracy. To double check if this is the actual plot, the code block was run few times, once even by restarting the runtime in Google Colab and every time, the same plot was found.

```
Computational complexity:       1.62 GMac
Number of parameters:           45.24 M
```

Figure 6: Model Complexity for VGG-19 Model

Figure 6 above shows the model complexity and total number of parameters count for the VGG-19 model.

To compare this model's results against VGG-16 and the baseline VGG model, it can clearly be concluded that this model fails in the training and validation process for the CIFAR-10 dataset. One of the reasons for this could be that this model is far more complicated and deeper for the simple, 3 channel CIFAR-10 dataset.

In terms of the model complexity and parameters count, as the number of fully connected layers increased, so did the model complexity and the total number of parameters count, which was expected from past model's results.

Problem 2:

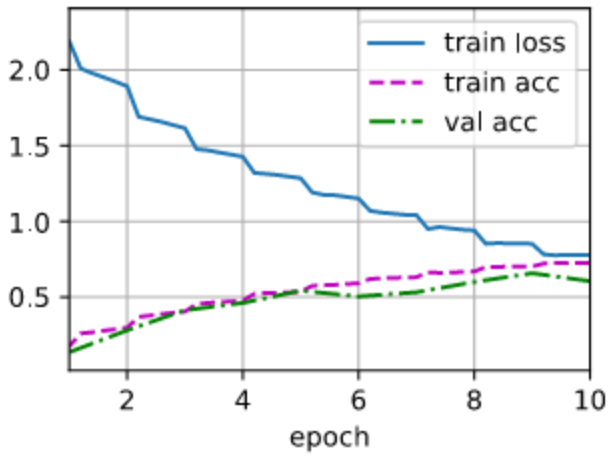

Figure 7: Results for Baseline GoogLeNet Model

Figure 7 above shows the plot which includes training loss, training accuracy and validation accuracy for the baseline GoogLeNet model. This model was trained on CIFAR-10 dataset, resizing all the images to 64x64 resolution over 10 Epochs. The training loss start out at around 2.0 and goes down to around 0.75. On the other hand, the training and validation accuracies start very low at around 0.2 and go up to around 0.75 too. For the most part, there is no over fitting, except between Epoch # 6 to 10, which is also not a lot.

```
Computational complexity:       129.76 MMac
Number of parameters:           5.98 M
```

Figure 8: Model Complexity for the Baseline GoogLeNet Model

Figure 8 above shows the model complexity and total number of parameters count for the baseline GoogLeNet model.

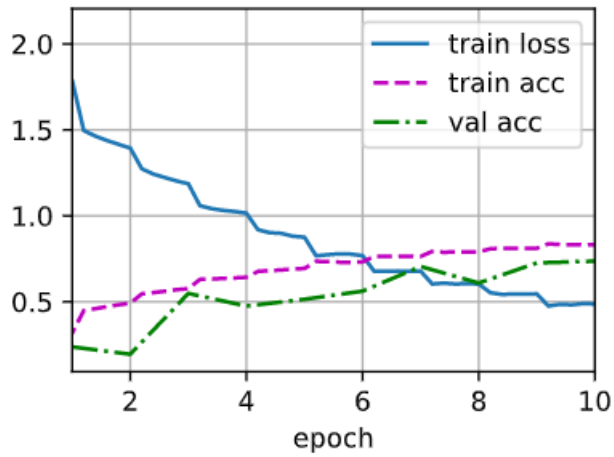

Figure 9: Results for the GoogLeNet Model with BatchNorm Added

Figure 9 above shows the plot which includes training loss, training accuracy and validation accuracy for the modified GoogLeNet model with Batch Normalization added. This model was trained on CIFAR-10 dataset, resizing all the images to 64x64 resolution over 10 Epochs. The training loss start out at around 1.75 and goes down to around 0.5. On the other hand, the training and validation accuracies start very low at around 0.2 and go up to around 0.75 too. For the most part, over fitting is visible.

```
Computational complexity:       130.02 MMac
Number of parameters:           6.0 M
```

Figure 10: Model Complexity for the GoogLeNet Model with BatchNorm Added

Figure 10 above shows the model complexity and total number of parameters count for the modified GoogLeNet model with Batch Normalization added.

To compare this modified model with Batch Normalization added with the baseline GoogLeNet model, this model performed better in some respects and worst in others. In terms of training loss, this model performed much better with lower loss values, while the accuracy values were around the same for both the models.

In terms of the model complexity, the computational complexity and total number of parameters count stayed almost the same.

Problem 3:

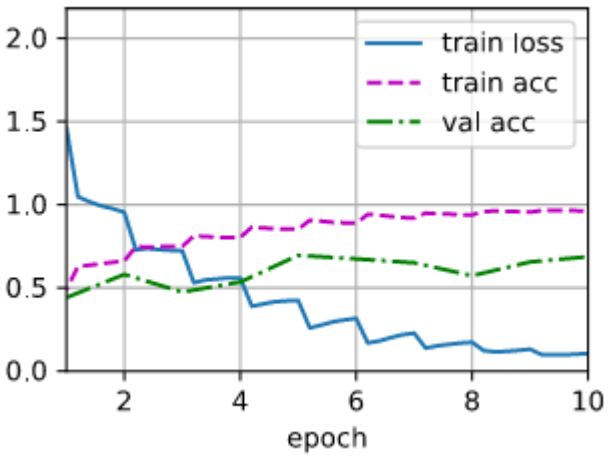

Figure 11: Results for ResNet-18 Model

Figure 11 above shows the plot which includes training loss, training accuracy and validation accuracy for ResNet-18 model. This model was trained on CIFAR-10 dataset, resizing all the images to 64x64 resolution over 10 Epochs. It can be seen that the training loss shows a good decreasing trend starting at around 1.5 and going down to around 0.1. On the other hand, the

training and validation accuracies also show a good trend by starting from 0.5 and going up to about 0.99 for the training dataset and to about 0.75 for the validation dataset. Starting from 1st Epoch, this model does show overfitting and the generalization gap is maintained throughout the plot.

```
Computational complexity:       148.76 MMac
Number of parameters:           11.18 M
```

Figure 12: Model Complexity for ResNet-18 Model

Figure 12 above shows the model complexity and total number of parameters count for ResNet-18 model.

To compare this model against GoogLeNet and VGG models, this model has shown the best results so far in terms of both the plot and the model complexity. The training loss value starts at a significantly lower point on the graph at around 1.5, which was above 2 for all the other, previous models. The training accuracy is also the best, going to about almost a 100%, which was not the case for any of the earlier models.

In terms of model complexity, this model's computational complexity is around the same as that of GoogLeNet and baseline VGG model and the total # of parameters count is the lowest of all the previous models.

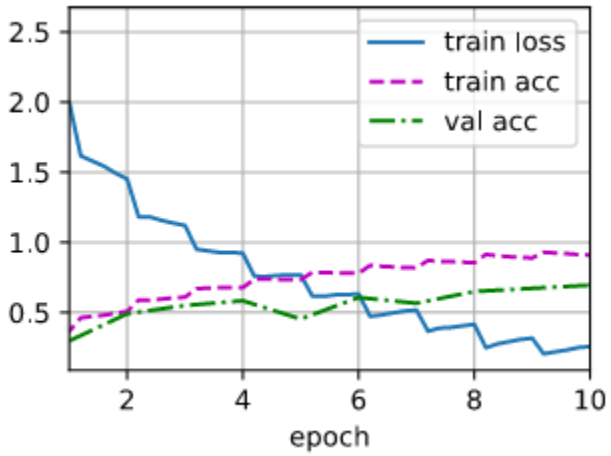

Figure 13: Results for ResNet-26 Model

Using the same, basic structure as ResNet-18 model, only this time adding few more Conv2d layers to create this ResNet-26 model with a total of 26 fully connected layers. Figure 13 above shows results for the same model which includes training loss, training accuracy and validation accuracy.

```
Computational complexity:       142.99 MMac
Number of parameters:           23.32 M
```

Figure 14: Model Complexity for ResNet-26 Model

Figure 14 above shows the model complexity and total number of parameters count for ResNet-26 model.

It can be seen that this model performs better than the ResNet-18 in some respects, while bad in other aspects. The training loss for this model starts at a much higher value than the earlier model, and it can also be seen that both training and validation accuracies are slightly less, but at the same time, the generalization gap is slightly smaller than the previous model, ResNet-18.

In terms of the model complexity, the computational complexity of ResNet-26 decreased, while the total number of parameters count increased, which was expected as the total number of fully connected layers increased.

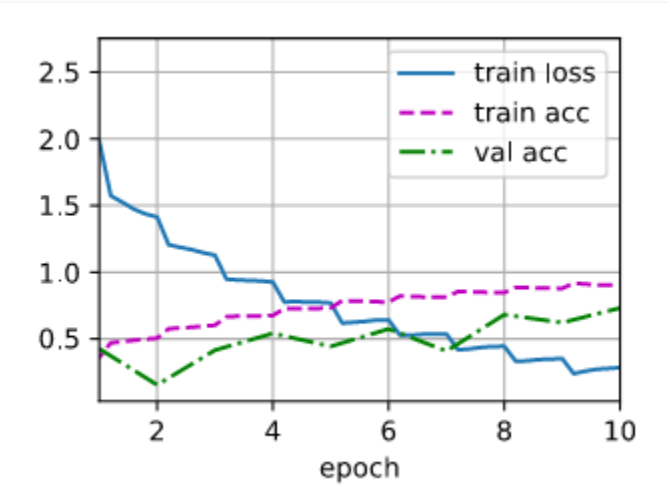

Figure 13: Results for ResNet-32 Model

```
Computational complexity:       86.41 MMac
Number of parameters:           23.78 M
```

Figure 16: Model Complexity for ResNet-32 Model

Figure 16 above shows the model complexity and total number of parameters count for ResNet-32 model.

In terms of the model complexity, the computational complexity of ResNet-32 decreased more while the total number of parameters count increased, which was expected as the total number of fully connected layers increased.

Bonus Problem:

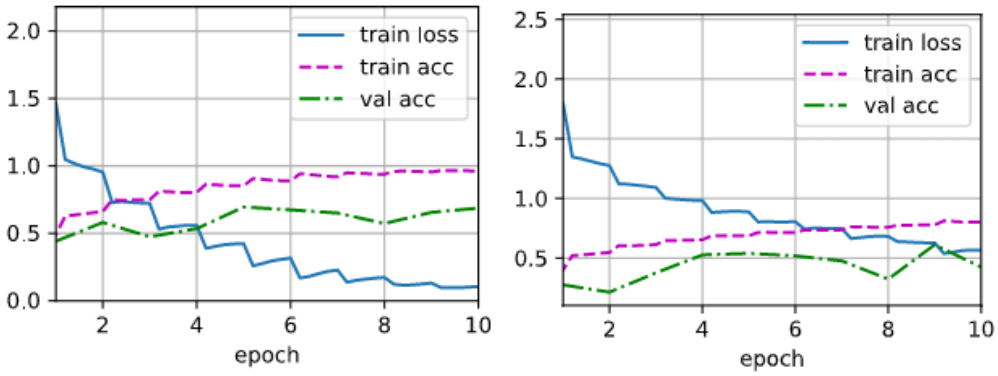

Figure 17: Results for ResNet-18 and DenseNet Model

For the bonus problem, using ResNet-18 as a base model, the DenseNet model was first plotted to check its Computational Complexity and Total Number of Parameters count. The goal of this problem was to tune the DenseNet model parameters for its computational complexity and total number of parameters count equals that of ResNet-18, while also moderately maintaining the loss and accuracy values. Figure 17 above shows the two plots, on the left is the ResNet-18 model, and on the right is the DenseNet model.

It can clearly be seen from the two plots that the training loss is approximately the same for both the models and show almost the same trend. But towards the end, ResNet-18 model did show lower loss values. On the other hand, the training and validation accuracies for both the models also show the same trend and have approximately the same values, but towards the end, ResNet-18 model did show better values.

Figure 18 below shows the computational complexity and total number of parameters count for ResNet-18 model.

```
Computational complexity:       148.76 MMac
Number of parameters:           11.18 M
```

Figure 18: Model Complexity for ResNet-18 Model

Figure 19 below shows the computational complexity and total number of parameters counts for the modified DenseNet model.

```
Computational complexity:       240.81 MMac
Number of parameters:           11.08 M
```

Figure 19: Model Complexity for Almost Equivalent DenseNet Model

To match the computational complexity and total number of parameters count of the DenseNet model, the number of output channels, kernel size, stride for the 1st Conv2d layer in 1st block were changed and same for the MaxPool layer. No specific method was followed, but different numbers based on educated guess were plugged in and experimented with. Next the growth rate was also increased.

The number of total parameters count has been almost matched for both the model, but the computational complexity for the DenseNet model was getting difficult to decrease as the number of parameters also changed. But, this number has been significantly brought down by tweaking the parameters as it was in Giga Macs, which has now been brought down to having only around 90 M of difference from the original ResNet-18 model.
