The Art & Science of Training Deep Neural Networks
CSCE 496/896

Programming Assignment 2

Spring 2021

Study of Two Convolutional Neural Network (CNN) Architectures: PlainNet & ResNet

---

CSCE 496: 80 points
CSCE 896: 130 points

Last Name 1:	Gao
First Name 1:	Tengjun(David)
NUID 1:	62915237
Last Name 2:	Choi
First Name 2:	Sung Woo
NUID 2:	79920240

Obtained Score:

496 & 896 PA2 Report

Gao Tengjun(David)
david.gao313@huskers.unl.edu

Sung Woo Choi
schoi9@huskers.unl.edu

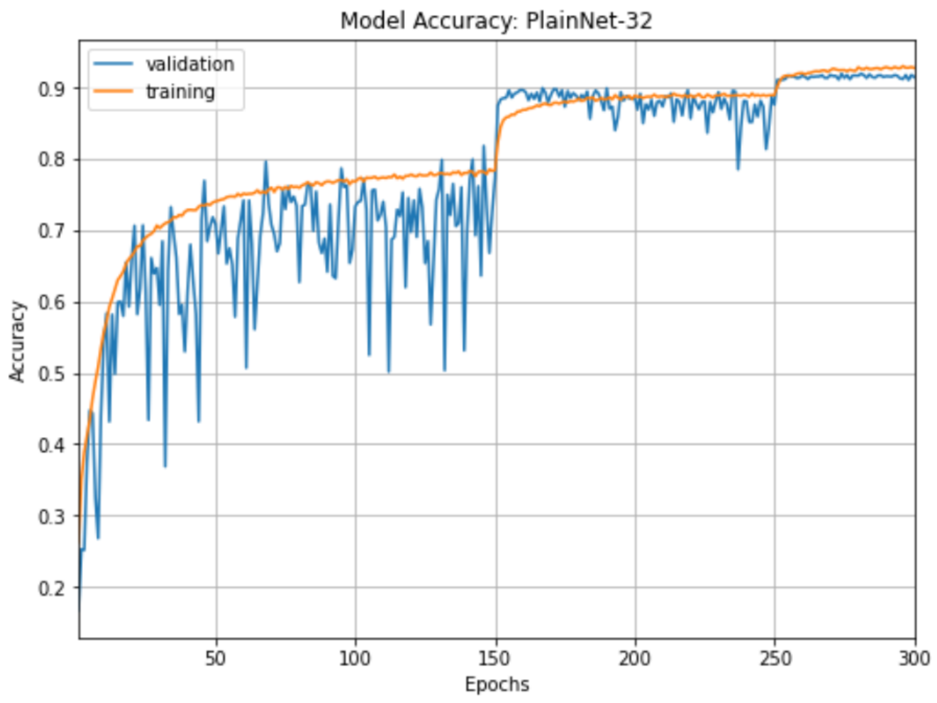
1/20/2021

1. PlainNet-32

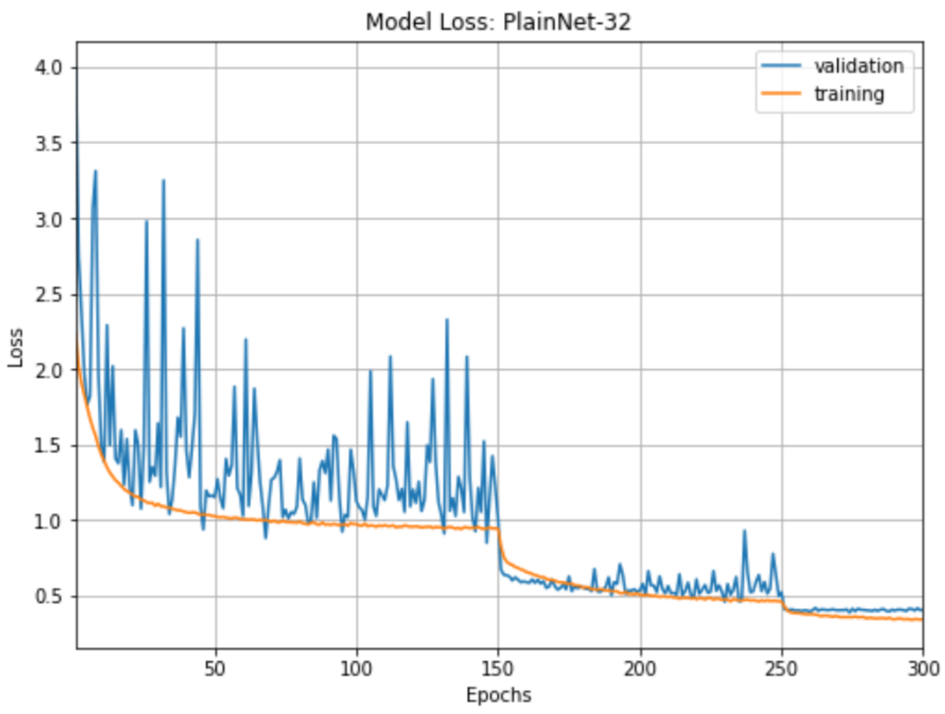
   (a) Train accuracy: 0.9641555547714233

   (b) Test accuracy: 0.9099000096321106

   (c) Learning curve (train & validation accuracy vs epochs)



(d) Learning curve (train & validation loss vs epochs)

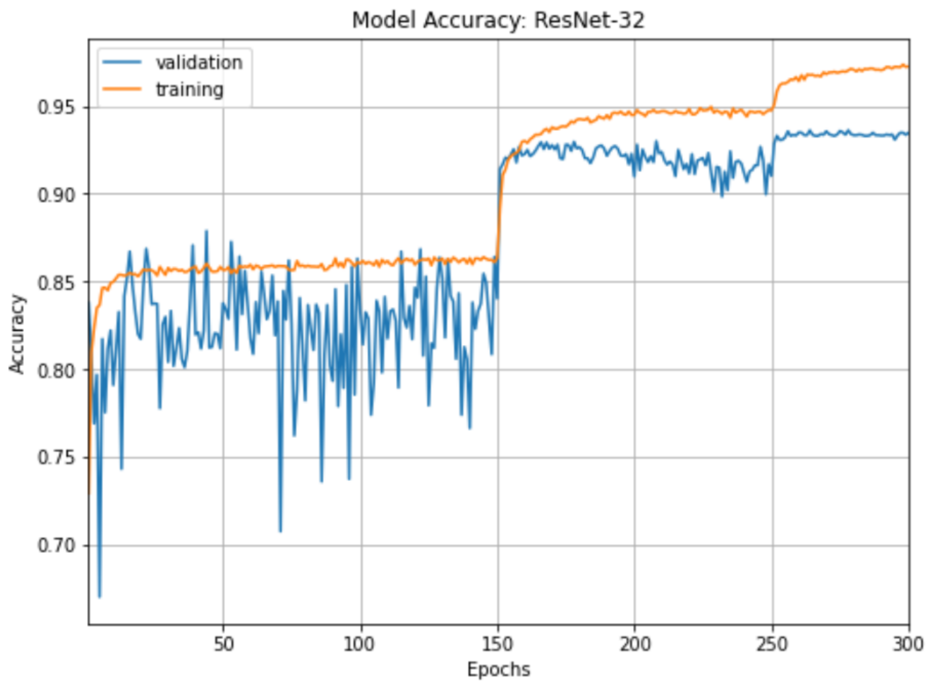


2. ResNet-32

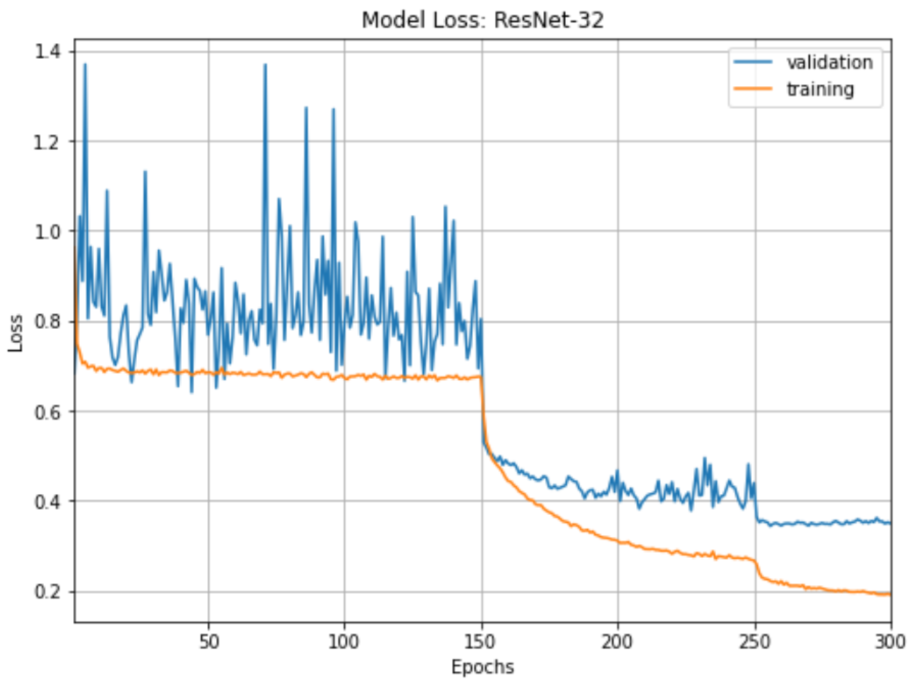
(a) Train accuracy: 0.9920222163200378

(b) Test accuracy: 0.9333999752998352

(c) Learning curve (train & validation accuracy vs epochs)



(d) Learning curve (train & validation loss vs epochs)

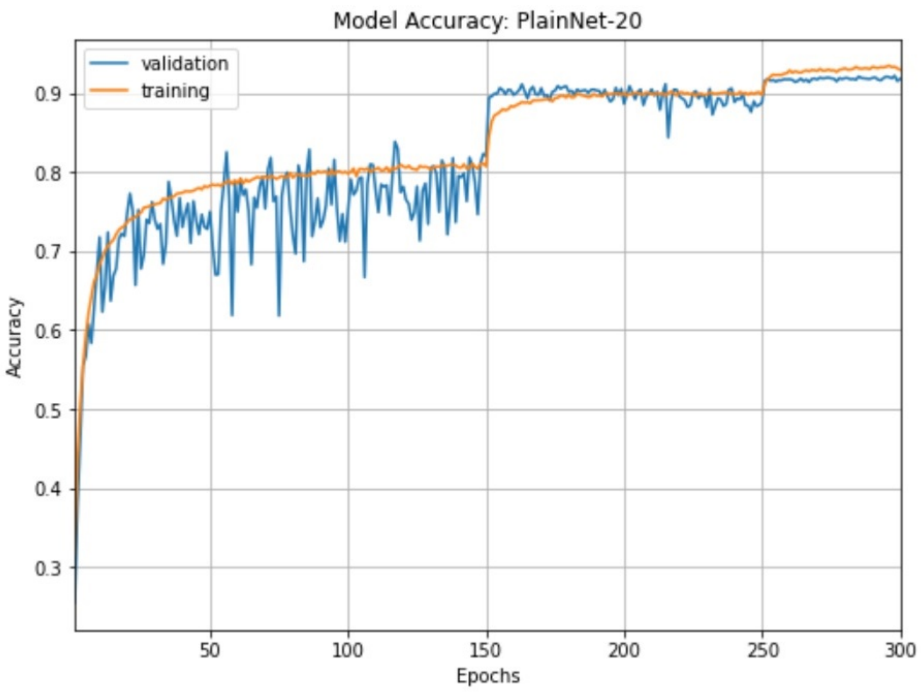


3. PlainNet-20

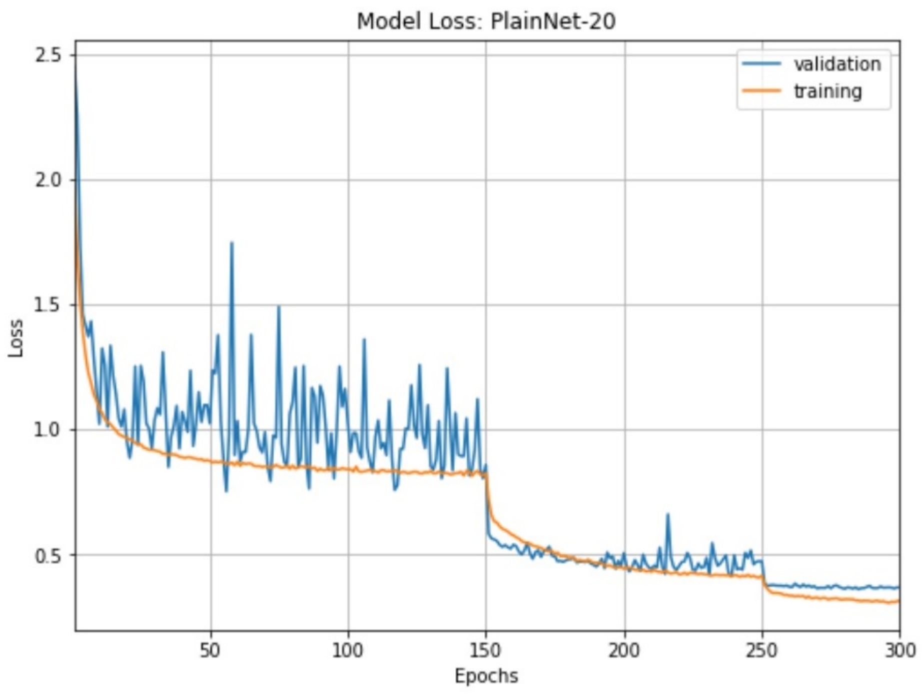
(a) Train accuracy: 0.9765111207962036

(b) Test accuracy: 0.9193000197410583

(c) Learning curve (train & validation accuracy vs epochs)



(d) Learning curve (train & validation loss vs epochs)



(c) Learning curve (train & validation accuracy vs epochs)

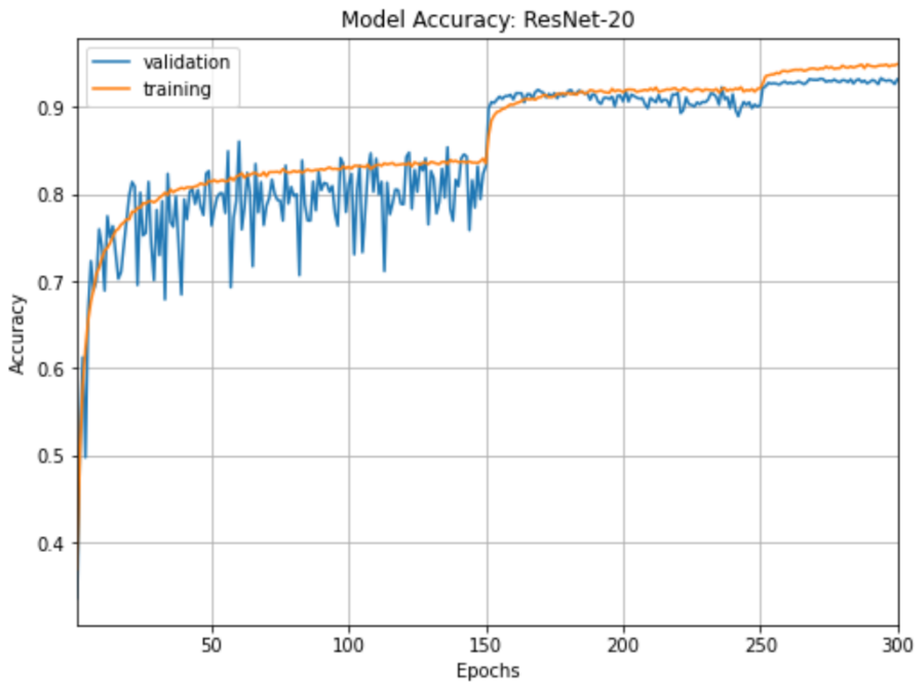
(d) Learning curve (train & validation loss vs epochs)

4. ResNet-20

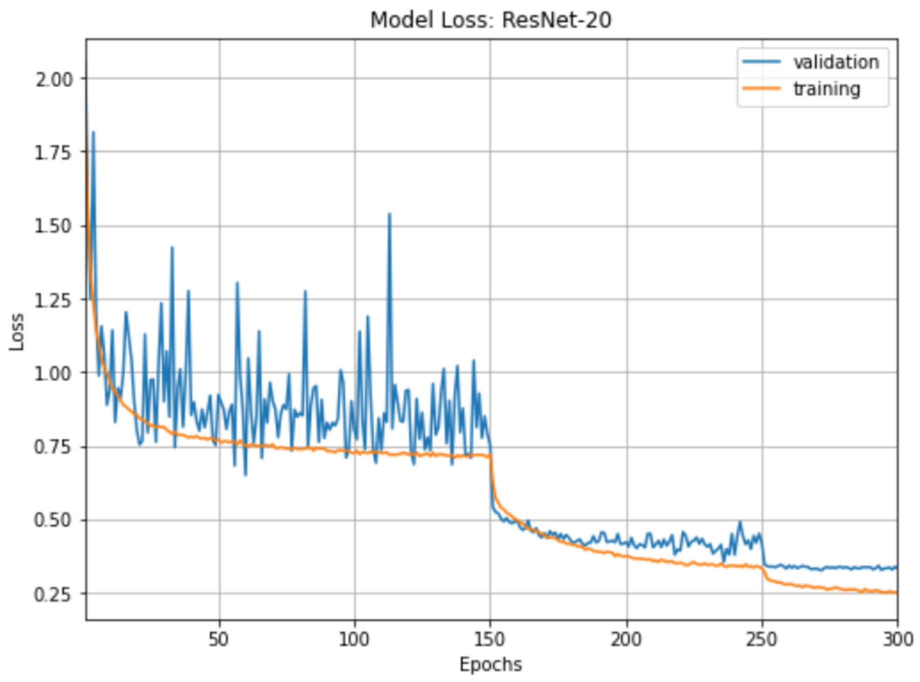
   (a) Train accuracy: 0.9765111207962036

   (b) Test accuracy: 0.9193000197410583

(c) Learning curve (train & validation accuracy vs epochs)



(d) Learning curve (train & validation loss vs epochs)



NOTE: For questions 2-4, we wanted to plot training accuracy learning curves of two models into a single graph, but the Colab was keep resetting and disconnecting in fact losing data of previous trained model due to long training time. As a result, we were unable to plot two training accuracy learning curves in a single graph.

- Q-1) Create a computational analysis table similar to AlexNet/VGGNet given in the following notebook for the ResNet-32 model you created. The table must have 4 columns: layer, output shape, memory, & parameters. Bias should be excluded from the computation. Last two rows should report total memory and total parameters. https://github.com/rhasanbd/Convolutional-Neural-Networks/blob/main/CNN-9-Notable%20Architectures-I.ipynb.

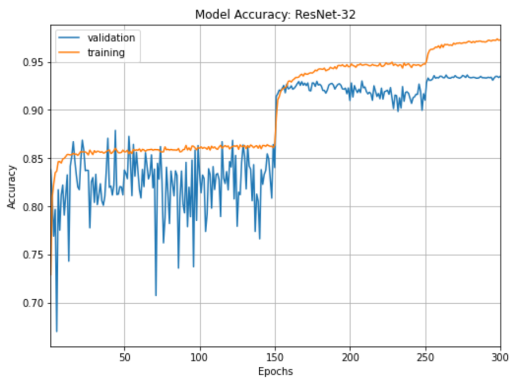
ResNet-32

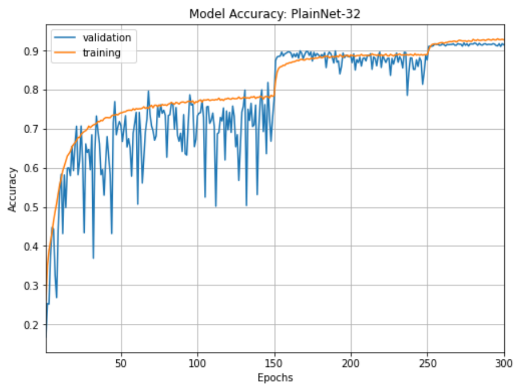
Layer	**Output Shape**	**Memory**	**Parameters**
INPUT:	[32 x 32 x 3]	32*32*3=3072	0
CONV 3 x 3 (16) stride 1, same padding	[32 x 32 x 16]	32*32*16=16384	(3*3*3)*16=432
BN	[32 x 32 x 16]	32*32*16=16384	4*16=64
ResNet Module 1			
ResNet Block 1	[32 x 32 x 16]	65536	4736
ResNet Block 2	[32 x 32 x 16]	65536	4736
ResNet Block 3	[32 x 32 x 16]	65536	4736
ResNet Block 4	[32 x 32 x 16]	65536	4736
ResNet Block 5	[32 x 32 x 16]	65536	4736
ResNet Module 2			
ResNet Block 1	[16 x 16 x 32]	49152	14720
ResNet Block 2	[16 x 16 x 32]	32768	18688
ResNet Block 3	[16 x 16 x 32]	32768	18688
ResNet Block 4	[16 x 16 x 32]	32768	18688
ResNet Block 5	[16 x 16 x 32]	32768	18688
ResNet Module 3			
ResNet Block 1	[8 x 8 x 64]	24576	58112
ResNet Block 2	[8 x 8 x 64]	16384	74240
ResNet Block 3	[8 x 8 x 64]	16384	74240
ResNet Block 4	[8 x 8 x 64]	16384	74240
ResNet Block 5	[8 x 8 x 64]	16384	74240
Global Average Pooling	[1 x 1 x 64]	64	0
Flatten	[1 x 1 x 64]	64	0
Output layer (include bias)	[1 x 1 x 10]	10	650
Total Memory		633994	
Total Parameters		469370	

ResNet-32: Calculation for the 1st ResNet Block of Each Module

Layer	**Output Shape**	**Memory**	**Parameters**
ResNet Module 1 (ResNet Block 1)			
CONV1 3 x 3 (16) stride 1, same padding	[32 x 32 x 16]	32*32*16=16384	(3*3*16)*16=2304
BN1	[32 x 32 x 16]	32*32*16=16384	4*16=64
CONV2 3 x 3 (16) stride 1, same padding	[32 x 32 x 16]	32*32*16=16384	(3*3*16)*16=2304
BN2	[32 x 32 x 16]	32*32*16=16384	4*16=64
ResNet Module 2 (ResNet Block 1)			
CONV1 3 x 3 (32) **stride 2**, same padding	[16 x 16 x 32]	16*16*32=8192	(3*3*16)*32=4608
BN1	[16 x 16 x 32]	16*16*32=8192	4*32=128
CONV2 3 x 3 (32) stride 1, same padding	[16 x 16 x 32]	16*16*32=8192	(3*3*32)*32=9216
BN2	[16 x 16 x 32]	16*16*32=8192	4*32=128
CONV3 1 x 1 (32) **stride 2**, same padding	[16 x 16 x 32]	16*16*32=8192	(1*1*16)*32=512
BN3	[16 x 16 x 32]	16*16*32=8192	4*32=128
ResNet Module 3 (ResNet Block 1)			
CONV1 3 x 3 (64) **stride 2**, same padding	[8 x 8 x 64]	8*8*64=4096	(3*3*32)*64=18432
BN1	[8 x 8 x 64]	8*8*64=4096	4*64=256
CONV2 3 x 3 (64) stride 1, same padding	[8 x 8 x 64]	8*8*64=4096	(3*3*64)*64=36864
BN2	[8 x 8 x 64]	8*8*64=4096	4*64=256
CONV3 1 x 1 (64) **stride 2**, same padding	[8 x 8 x 64]	8*8*64=4096	(1*1*32)*64=2048
BN3	[8 x 8 x 64]	8*8*64=4096	4*64=256

- Q-2) Why is the performance of the ResNet-32 model better than the PlainNet-32 model? Explain why PlainNet-32 performed poorly (both during training & testing). If you create a deeper PlainNet would its performance improve?





ResNet-32 - Train Accuracy: 0.9920222163200378
ResNet-32 - Test Accuracy: 0.9333999752998352
ResNet-32 - Train Loss: 0.137788787484169
ResNet-32 - Test Loss: 0.35249197483062744

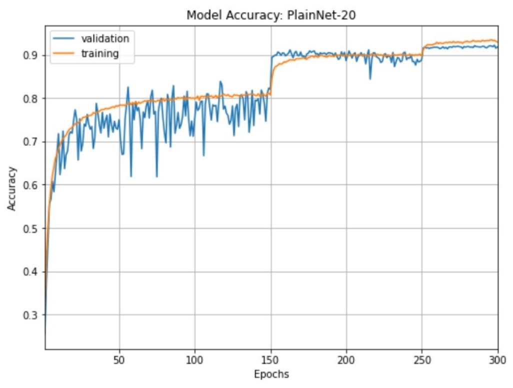
PlainNet-32 - Train Accuracy: 0.9641555547714233
PlainNet-32 - Test Accuracy: 0.9099000096321106
PlainNet-32 - Train Loss: 0.24021343886852264
PlainNet-32 - Test Loss: 0.43544721603393555

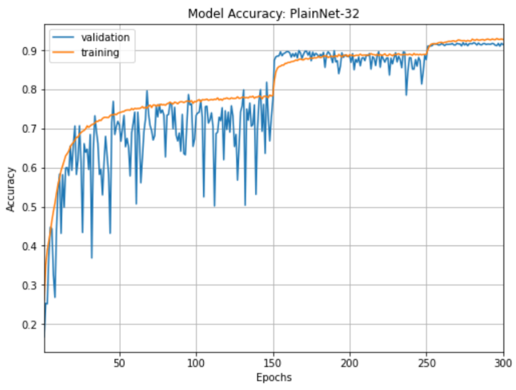
The performance of the ResNet-32 model is better than the PlainNet-32 model during both training and testing because ResNet-32 has skip connections. The advantages of the skip connections are that if weights are close to zero, signals will pass through skip connections instead of the residual blocks so that signals can propagate back and forth through the network, and weights can learn more data without losing useful information.

PlainNet-32 preformed poorly during both training and testing because the training error increases as the signal passes through to deeper network and the weights in PlainNet-32 do not learn as effectively compared to those in ResNet-32 due to lack of skip connections. Thus, the deeper PlainNet would not improve the performance.

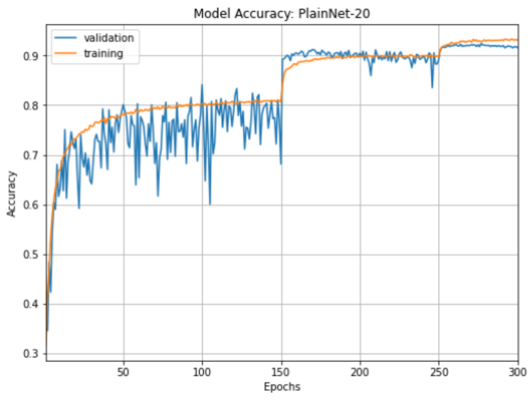
- Q-3) Compare the training accuracy of PlainNet-20 and the PlainNet-32 models. Using training accuracy learning curves argue which model has higher training accuracy & why? Which model has higher test accuracy and why?

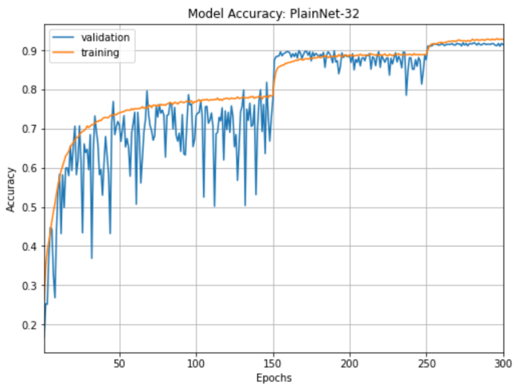
(1) Test result in our Jupiter network:





(2) Previous test learning curves not in Jupiter network.





Final Results of test train accuracy and train & test loss:

PlainNet-32 - Train Accuracy: 0.9641555547714233
PlainNet-32 - Test Accuracy: 0.9099000096321106
PlainNet-32 - Train Loss: 0.24021343886852264
PlainNet-32 - Test Loss: 0.43544721603393555

(1) Test result in our Jupiter network:
PlainNet-20 - Train Accuracy: 0.9681333303451538
PlainNet-20 - Test Accuracy: 0.9154999852180481
PlainNet-20 - Train Loss: 0.20892927050590515
PlainNet-20 - Test Loss: 0.38632771372795105

(2) Previous test result not in Jupiter network:
PlainNet-20 - Train Accuracy: 0.9639333486557007
PlainNet-20 - Test Accuracy: 0.9097999930381775
PlainNet-20 - Train Loss: 0.22100095450878143
PlainNet-20 - Test Loss: 0.39989858865737915

Compare to PlainNet-20, PlainNet-32 has four more convolution layers for each 'blocks'.

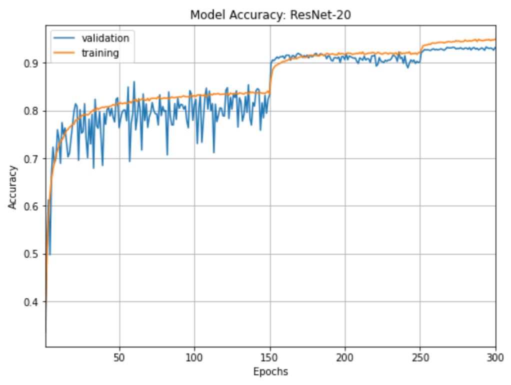
For PlainNet-20, we have two test results(marked '(1)' '(2)'): one in Jupiter network and the other one that got erased after new training. The test results are different because of data augmentation. We resized and randomly cropped, horizontally flipped, and zoomed our testing data.

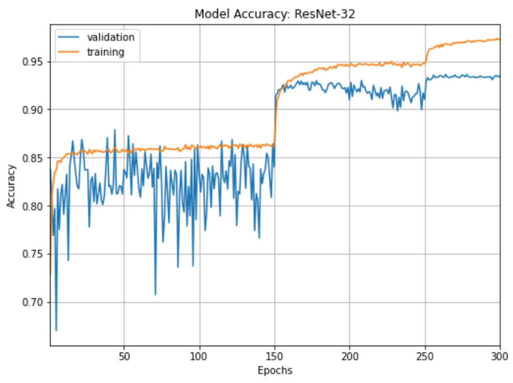
Following observations are based on the test accuracy. Based on (1) test result, the test accuracy of PlainNet-20 was slight bit better than the test accuracy of PlainNet-32. However, based on (2) test result, the test accuracy of PlainNet-20 was slight bit worse than the test accuracy of PlainNet-32. The difference between the test accuracy of two models are around 0.5 percentage for both testings (the error range between the test accuracy of two models are around $\pm 5\%$). This means the the test accuracy for PlainNet-32 and PlainNet-20 are very close to each other. Thus, deeper network did not improve test accuracy. This is because of the degradation problem.

Following observations are based on learning curves of training accuracy. Based on the experiments, the learning curve of PlainNet-20 has higher training accuracy compared to the learning curve of the PlainNet-32. This means that deeper network has the learning curve with lower training accuracy. This is because PlanNet-32 is facing the degradation problem that causes training errors. The degradation problem happens on training accuracy as the network depth increases because the training accuracy gets saturated and then degrades rapidly.

As a conclusion, the PlainNet-20 has higher training accuracy and similar test accuracy compared to the PlainNet-32 because of degradation problem in PlanNet-32.

- Q-4) Compare the training accuracy of ResNet-20 and the ResNet-32 models. Using training accuracy learning curves argue which model has higher training accuracy & why? Which model has higher test accuracy and why?





ResNet-20 - Train Accuracy: 0.9765111207962036
ResNet-20 - Test Accuracy: 0.9193000197410583
ResNet-20 - Train Loss: 0.1731833815574646
ResNet-20 - Test Loss: 0.37651029229164124

ResNet-32 - Train Accuracy: 0.9920222163200378

```
ResNet-32 - Test Accuracy: 0.9333999752998352
ResNet-32 - Train Loss: 0.137788787484169
ResNet-32 - Test Loss: 0.35249197483062744
```

Based on the experiment, the ResNet-32 model has higher training accuracy and higher test accuracy compared to the ResNet-20 model. This means that the ResNet-32, the deeper network, can handle the degradation problem well by using skip connection. The advantages of the skip connections are that if weights are close to zero, signals will pass through skip connections instead of the residual blocks so that signals can propagate back and forth through the network, and weights can learn more data without losing useful information. In fact, the skip connection reduces the training error that is caused by large depth of the network.