ECE 361E: Homework 4

Sidharth Babu, SNB2593 Tianda Huang, TH32684

March 8, 2023

Problem 1

Question 2

Figure 1: *Table 1*

Model	Training Accuracy [%]	Test Accuracy [%]	Total Time for Training [s]	Number of Trainable Params
VGG11	99.69	78.25	773.210	9,750,922
VGG16	99.69	77.64	1223.664	14,655,050
MobileNet	99.30	79.41	1186.733	3,217,226

Question 3

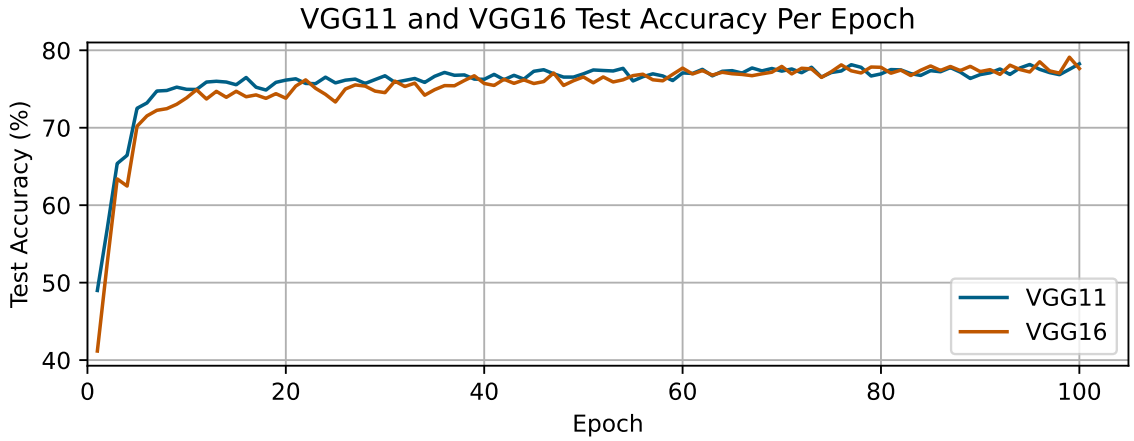

Figure 2: Test Accuracy of VGG11 and VGG16

Problem 2

Question 2

Figure 3: *Table 2*

	Total Inference Time [s]		RAM memory [MB]		Accuracy [%]	
	MC1	RPi	MC1	RPi	MC1	RPi
VGG11	648.79	582.19	306	155	78.25	78.25
VGG16	1063.83	1004.81	325	174	77.64	77.64
MobileNet	495.46	199.86	282	128	79.41	79.41

Question 3

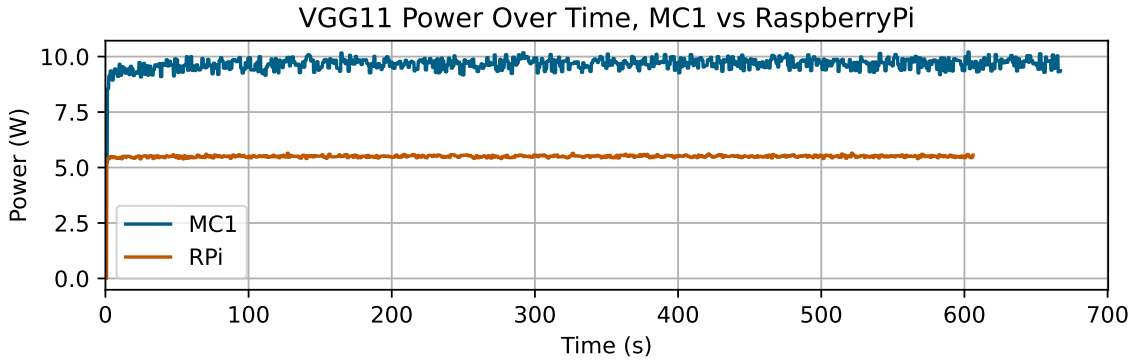

Figure 4

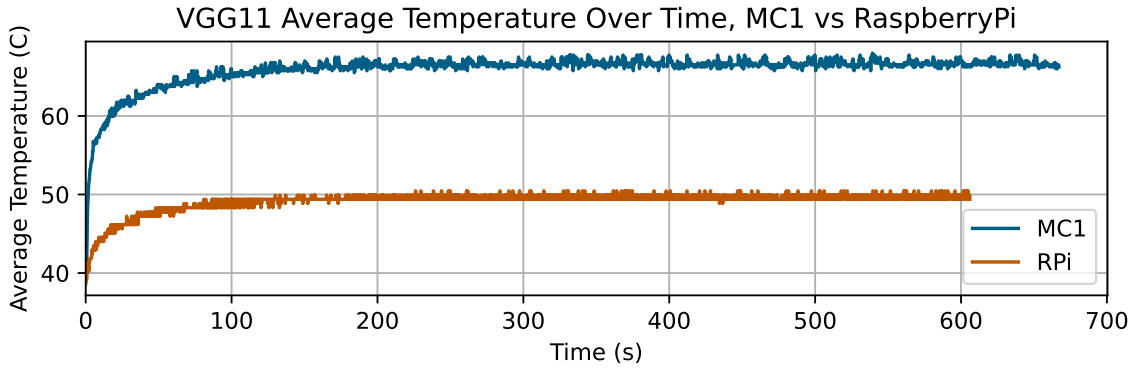

Figure 5

Figure 6: *Table 3*

Model	MC1 Total Energy Consumption [J]	RPi total Energy Consumption [J]
VGG11	6442.54	3330.53
VGG16	10847.49	5675.75
MobileNet	3957.77	1168.07

Problem 3

BONUS

In order to properly discuss the tradeoffs between the two deployment frameworks, first we must include the data that we profiled with the same models on ONNX.

Figure 7: *Table 4 - ONNX Model Summary*

Model	Training Accuracy [%]	Test Accuracy [%]	Total Time for Training [s]	Number of Trainable Params
VGG11	97.57	76.48	3011.79	9,750,922
VGG16	97.86	78.89	3622.42	14,655,050
MobileNet	99.42	77.75	2211.56	3,217,226

Figure 8: *Table 5 - ONNX Deployment Summary*

	Total Inference Time [s]		RAM memory [MB]		Accuracy [%]	
	MC1	RPi	MC1	RPi	MC1	RPi
VGG11	658.23	680.61	330	171	76.48	76.48
VGG16	990.92	1172.02	352	192	78.89	78.89
MobileNet	491.65	329.30	302	139	77.75	77.75

Figure 9: *Table 6 - ONNX Energy Summary*

Model	MC1 Total Energy Consumption [J]	RPi total Energy Consumption [J]
VGG11	6574.78	3739.40
VGG16	10106.79	6381.64
MobileNet	4196.58	1877.45

Overall, the TF deployment is more efficient based on our data.

Looking at subsection 1 of tables 5 and 2, we can see that the TF Deployment has lower inference times than the ONNX deployment in all cases except for VGG16-MC1 and MobileNet-MC1. This could be due to some particular hardware feature of the MC1 that the ONNX framework optimizes better for. It could also be spurious data, and repeating this experiment may be necessary to confirm this either way.

In subsection 2 of tables 5 and 2, we can see that the TF deployment uses less RAM in all cases. This could be due to a variety of reasons, but could indicate that TF's memory footprint is generally be

better than ONNX's. In order to properly confirm this, additional models would have to be tested on both frameworks, and on more hardware platforms.

In tables 3 and 6, we can see that the TF deployment uses less energy in all cases besides the VGG16-MC1. Since this is an outlier, it may be that this data point is spurious, but it would need to be confirmed through additionalruns of the experiment.

With these comparisons on our data, we can see that Tensorflow Lite seems to perform better in all three key metrics. Therefore, we would claim that Tensorflow is the more efficient deployment framework.

In terms of preference, the ONNX and TFLITE frameworks both present their own advantages. ONNX may generally be better for R&D purposes, as it is more flexible, and does not restrict the user to a particular model training framework. However, for production deployment at scale, TFLITE is likely the better choice, as we have shown that it is generally more efficient.

Contributions and Valuable Things Learned

Both group members, Sidharth Babu and Tianda Huang, contributed an equal amount of work due to working on the entire project together. We learned about the differences between Tensorflow and Pytorch to develop the models, and we learned about the differences between ONNX and Tensorflow for deployment.