Roofline Modeling

ImageNet Analysis

Name: Mihir Prajapati

1. Experiment Design

The main aim of this project is to check the performance of different GPUs on the cloud when training the ImageNet dataset using different architectures. As this is just a test of GPU performance, the whole dataset is not needed, and the model also does need not to be trained all the way through.

So, for the purpose of this experiment the following GPUs were used on the NYU HPC provided:

- **A100 (GPU: 1)**
- **V100 (GPU: 1)**

And the following architectures were trained:

- **resnet18**
- **alexnet**

As the focus of the experiment is just GPU performance, the dataset that was used was just a **dummy dataset** that randomly generated using **FakeData** function provided by PyTorch. It had the same input size as the ImageNet dataset, so it doesn't have any impact on the expected output as compared to the original dataset.

2. Complexity Estimation and Measurement

To measure performance, two things are estimated the **FLOPs** during training and the **bytes** used.

For this, the **profiler** from the **torch** library is utilized in conjunction Nsight profiler command **ncu** in the terminal to run the python script. The profiling is done using the commands **ncu.start()** and **ncu.stop()**, which are inserted at points in the **train** function where FLOPs estimation is required.

The following command was used to obtain all required metrics to calculate **FLOPs** and **memory**:

```
ncu -f --log-file resnet18_A100.log --profile-from-start off --replay-mode application --metrics
smsp_sass_thread_inst_executed_op_fp16_pred_on.sum,smspsass_thread_inst_executed_op_fadd_pre
d_on.sum,smspsass_thread_inst_executed_op_fmul_pred_on.sum,smspsass_thread_inst_executed_op
_ffma_pred_on.sum,dramsectors_write.sum,drambytes_write.sum.per_second,dramsectors_read.sum,d
ram_bytes_read.sum.per_second --target-processes all python3 ./main.py --batch-size 4 --epochs 1 --
print-freq 10 -a resnet18 --dummy
```

The following shell script command was used to retrieve the records from the generated LOGs:

```
ncu -f --log-file $LOG --metrics $METRICS --target-processes all $RUN_COMMAND

for TYPE in ${METRICS//,/ } ; do
 cat $LOG | grep $TYPE | sed -e "s/,//g" | awk -v t="$TYPE"
'BEGIN{sum=0}{sum=sum+$3}END{printf("%s %d\n", t, sum);}' >> $ LOG done
```

FLOPs were calculated using the following formula:

Roofline Modeling

ImageNet Analysis

Name: Mihir Prajapati

FLOPs = smspsass_thread_inst_executed_op_fadd_pred_on.sum + smspsass_thread_inst_executed_op_fmul_pred_on.sum + (smspsass_thread_inst_executed_op_ffma_pred_on.sum * 2)

Bytes accessed:

(drambytes_write.sum.per_second + dramsectors_read.sum) * 32

Arithmetic Intensity: FLOPs / Bytes

Attainable FLOPs/sec: FLOPs/ total time

The following values were obtained for each architecture on each GPU:

ResNet-18 Model

NVIDIA A100 GPU
gpu__time_duration.sum: 4953 ms
dram__bytes_read.sum: 67,685
dram__bytes_write.sum: 10,175
dram__sectors_write.sum: 721,025
dram__sectors_read.sum: 32,093,153
smsp__sass_thread_inst_executed_op_fadd_pred_on.sum: 132,532,938
smsp__sass_thread_inst_executed_op_fmul_pred_on.sum: 169,224,032
smsp__sass_thread_inst_executed_op_ffma_pred_on.sum: 139,482,243
Total FLOPs: 580,721,456
Arithmetic Intensity (AI): 0.553 FLOPS/byte
Performance: 0.1172 TFLOPS

NVIDIA V100 GPU
gpu__time_duration.sum: 6892 ms
dram__bytes_read.sum: 38,461
dram__bytes_write.sum: 26,543
dram__sectors_write.sum: 14,217,318
dram__sectors_read.sum: 27,932,573
smsp__sass_thread_inst_executed_op_fadd_pred_on.sum: 598,970,432
smsp__sass_thread_inst_executed_op_fmul_pred_on.sum: 126,847,035
smsp__sass_thread_inst_executed_op_ffma_pred_on.sum: 15,498,365,832
Total FLOPs: 31,722,549,131
Arithmetic Intensity (AI): 23.5191 FLOPS/byte
Performance: 4.602 TFLOPS

AlexNet Model

NVIDIA A100 GPU
gpu__time_duration.sum: 2453 ms
dram__bytes_read.sum: 19,034

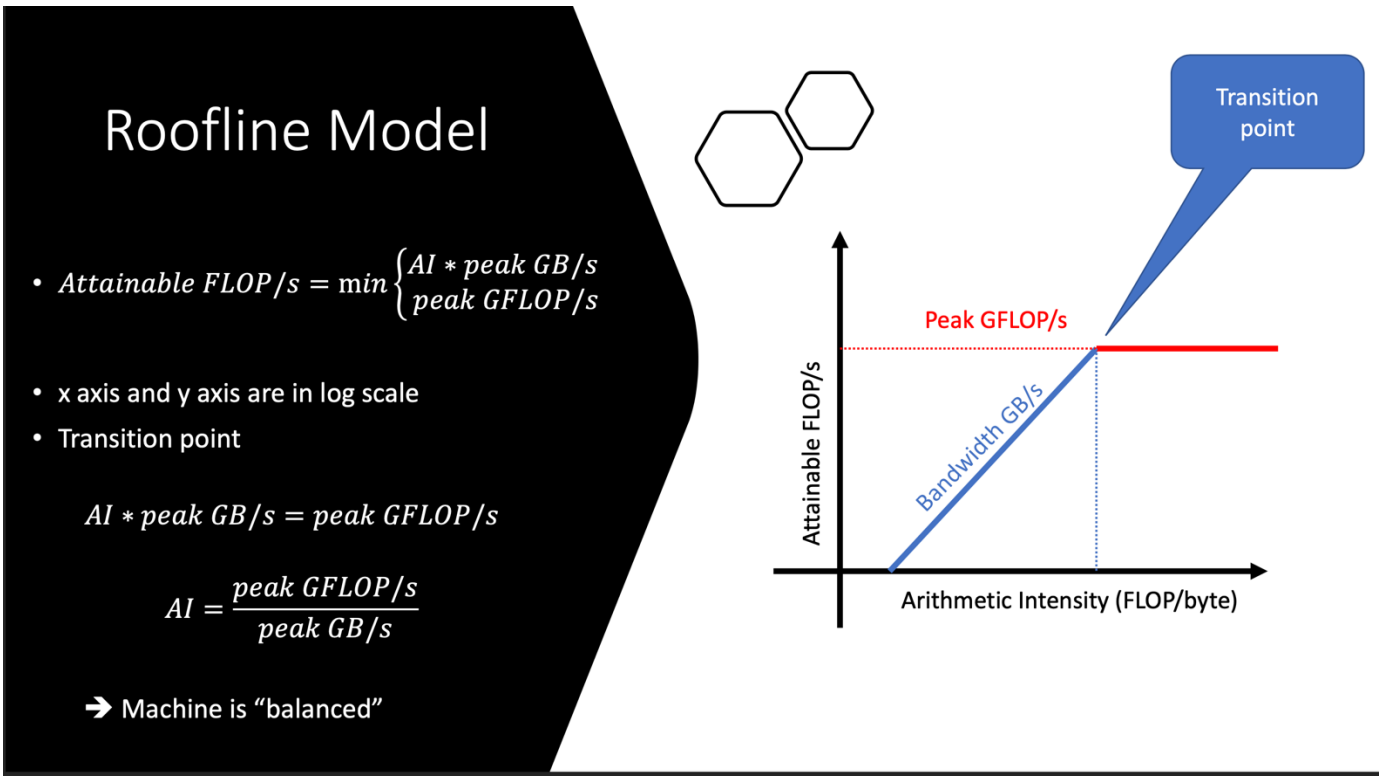
Roofline Modeling

ImageNet Analysis

Name: Mihir Prajapati
dram__bytes_write.sum: 5,179
dram__sectors_write.sum: 6,132,105
dram__sectors_read.sum: 19,934,612
smsp__sass_thread_inst_executed_op_fadd_pred_on.sum: 25,201,689
smsp__sass_thread_inst_executed_op_fmul_pred_on.sum: 112,320,861
smsp__sass_thread_inst_executed_op_ffma_pred_on.sum: 2,115,336,075
Total FLOPs: 4,368,194,700
Arithmetic Intensity (AI): 5.2367 FLOPS/byte
Performance: 1.780 TFLOPS

NVIDIA V100 GPU
gpu__time_duration.sum: 3531 ms
dram__bytes_read.sum: 14,752
dram__bytes_write.sum: 8,137
dram__sectors_write.sum: 14,491,783
dram__sectors_read.sum: 24,013,751
smsp__sass_thread_inst_executed_op_fadd_pred_on.sum: 181,164,298
smsp__sass_thread_inst_executed_op_fmul_pred_on.sum: 147,095,735
smsp__sass_thread_inst_executed_op_ffma_pred_on.sum: 6,092,243,504
Total FLOPs: 12,512,747,041
Arithmetic Intensity (AI): 10.3799 FLOPS/byte
Performance: 3.543 TFLOPS

3. Roofline Modeling and Discussion

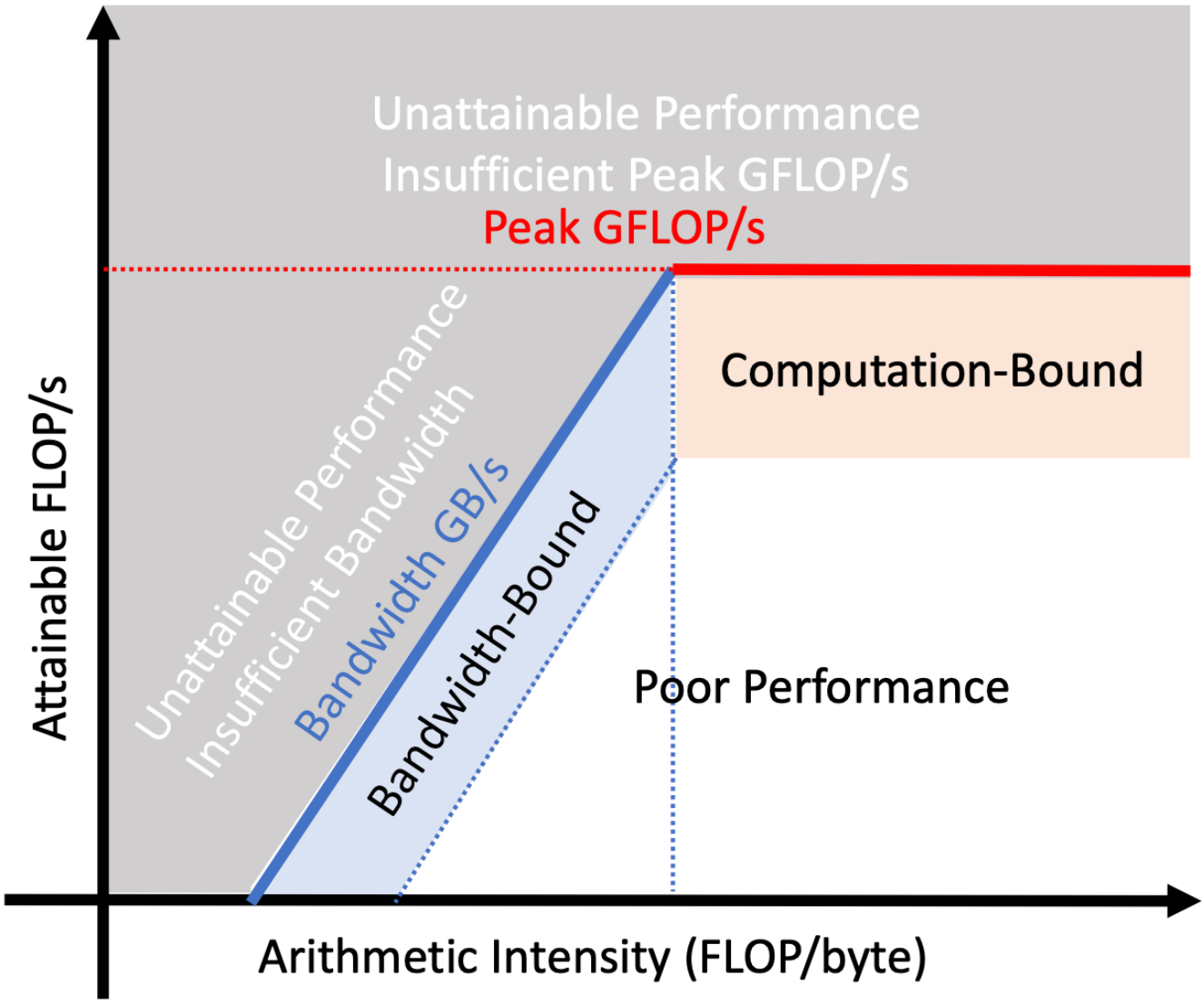


Roofline Modeling

ImageNet Analysis

Name: Mihir Prajapati

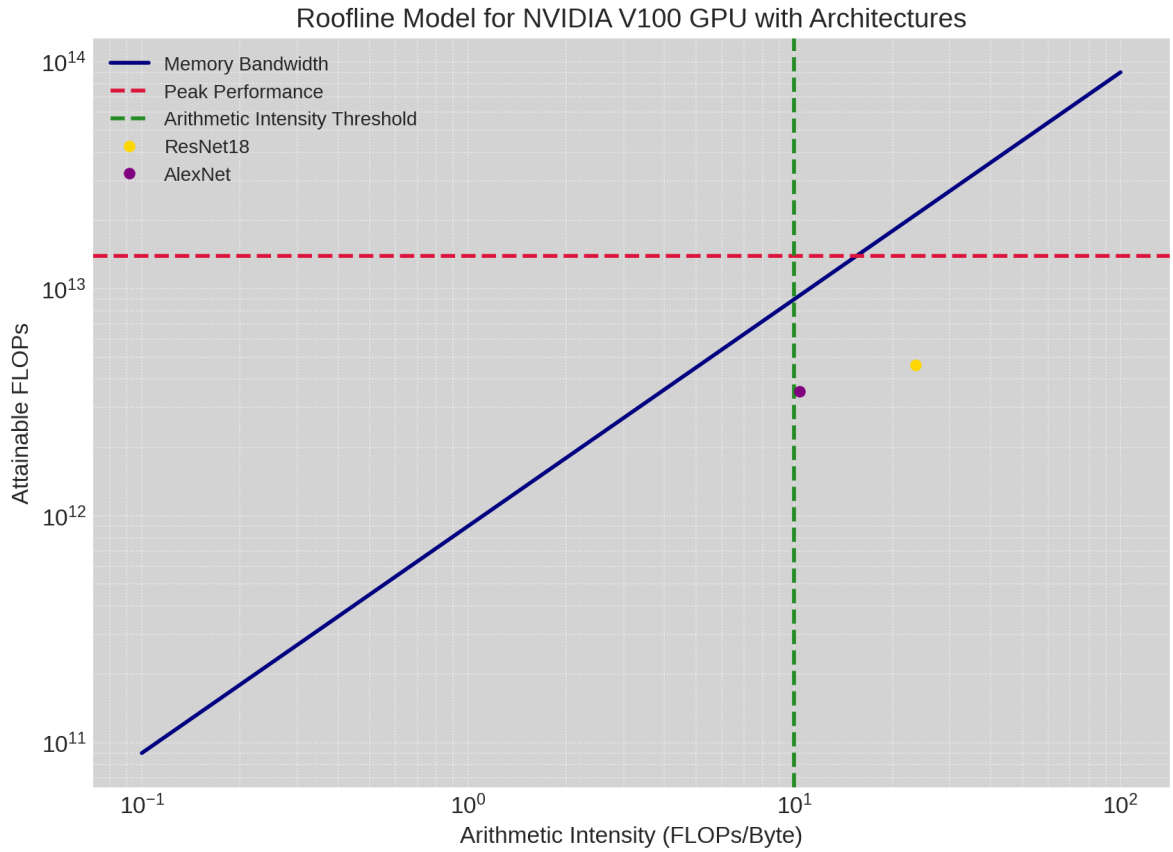
By referring the following graph, discussion can be done of an architecture's performance on a GPU.



Roofline Modeling

ImageNet Analysis

Name: Mihir Prajapati



The horizontal dashed red line is the max FLOPs of the GPU which is 14 TFLOPs.

For ResNet18, a relatively low AI and attainable TFLOPs, this point indicates that the V100 is not being fully utilized for ResNet18. This suggests that ResNet18 may be memory-bound on the V100, and the performance is likely limited by memory bandwidth.

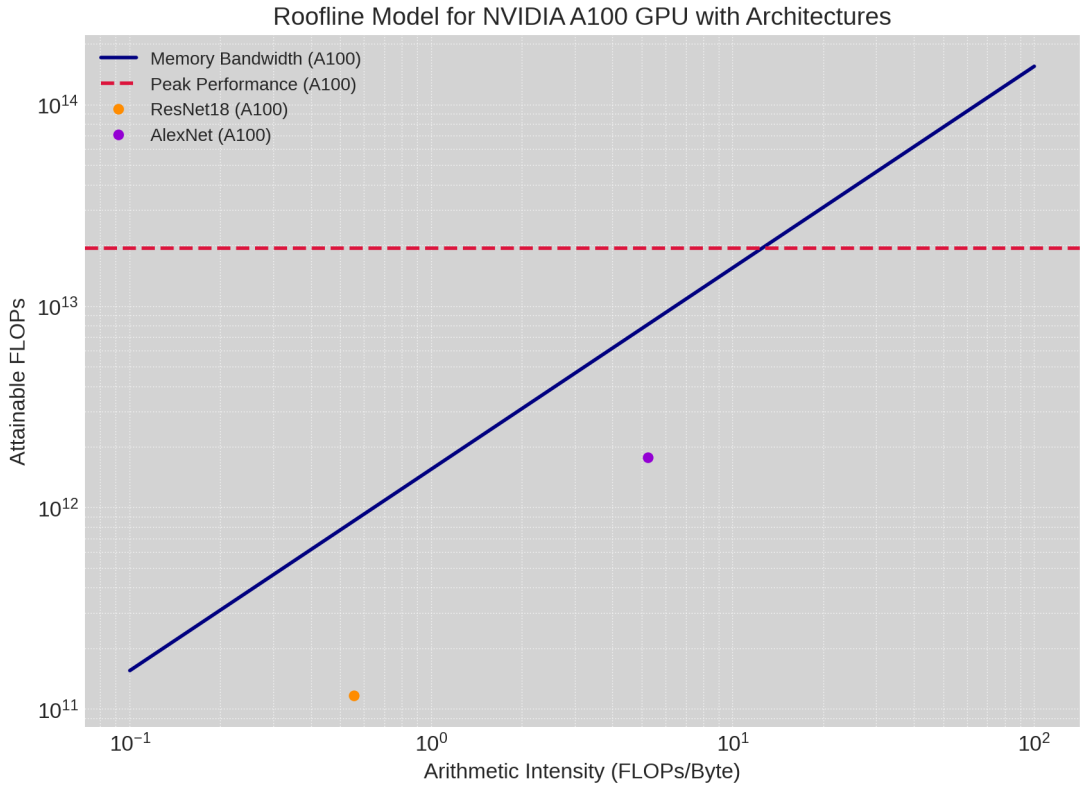
AlexNet on V100 appears to be slightly more compute-intensive than ResNet18 with a higher AI, but still does not come close to the peak performance of the V100. It indicates that while AlexNet is utilizing the GPU's resources better than ResNet18, there's still a considerable gap to reach peak performance, suggesting that there is room for optimization.

Both architectures are well below the peak performance line, indicating that neither is fully utilizing the compute potential of the V100 GPU.

Roofline Modeling

ImageNet Analysis

Name: Mihir Prajapati



ResNet18 has a lower arithmetic intensity and FLOPs, it's less compute-intensive and more memory bandwidth-bound. Its performance is closer to the bandwidth line, suggesting memory accesses are a limiting factor.

AlexNet achieves higher TFLOPS and has a higher arithmetic intensity than ResNet18, suggesting it is making better use of the A100 GPU's compute resources, but it is still below the peak performance, possibly due to a combination of compute and memory limitations.

AlexNet appears to be more compute-bound than ResNet18, given its higher position on the graph relative to the memory bandwidth line.

4. Conclusion

Thus, it is clear that different performance is obtained on each of the GPUs for different architectures. AlexNet has better performance on the A100, and ResNet18 seems to perform better on the V100.

References

[1] https://github.com/pytorch/examples/tree/master/imagenet
[2] https://developer.nvidia.com/nsight-systems