Name: *Ruiqi Li*
NetID: *Ruiqili4*
Section: *AL1*

ECE 408/CS483 Milestone 2 Report

1. Show output of rai running Mini-DNN on the basic GPU convolution implementation for batch size of 1k images. This can either be a screen capture or a text copy of the running output. Please do not show the build output. (The running output should be everything including and after the line "*Loading fashion-mnist data...Done*").

<output here>

```
Loading model...Done
Conv-GPU==
Layer Time: 100.138 ms
Op Time: 4.65247 ms
Conv-GPU==
Layer Time: 96.732 ms
Op Time: 29.7178 ms

Test Accuracy: 0.886


real    0m9.782s
user    0m9.404s
sys     0m0.304s
```

2. For the basic GPU implementation, list Op Times, whole program execution time, and accuracy for batch size of 100, 1k, and 10k images.

Batch Size	Op Time 1	Op Time 2	Total Execution Time	Accuracy
100	*0.48717 ms*	*2.93471 ms*	*0m1.287s*	*0.86*
1000	4.65247 ms	29.7178 ms	0m9.782s	0.886
10000	*44.8455 ms*	*276.037 ms*	*1m38.762s*	*0.8714*

3. List all the kernels that collectively consumed more than 90% of the kernel time and what percentage of the kernel time each kernel did consume (start with the kernel that consumed the most time, then list the next kernel, until you reach 90% or more).

conv_forward_kernel – 100% time

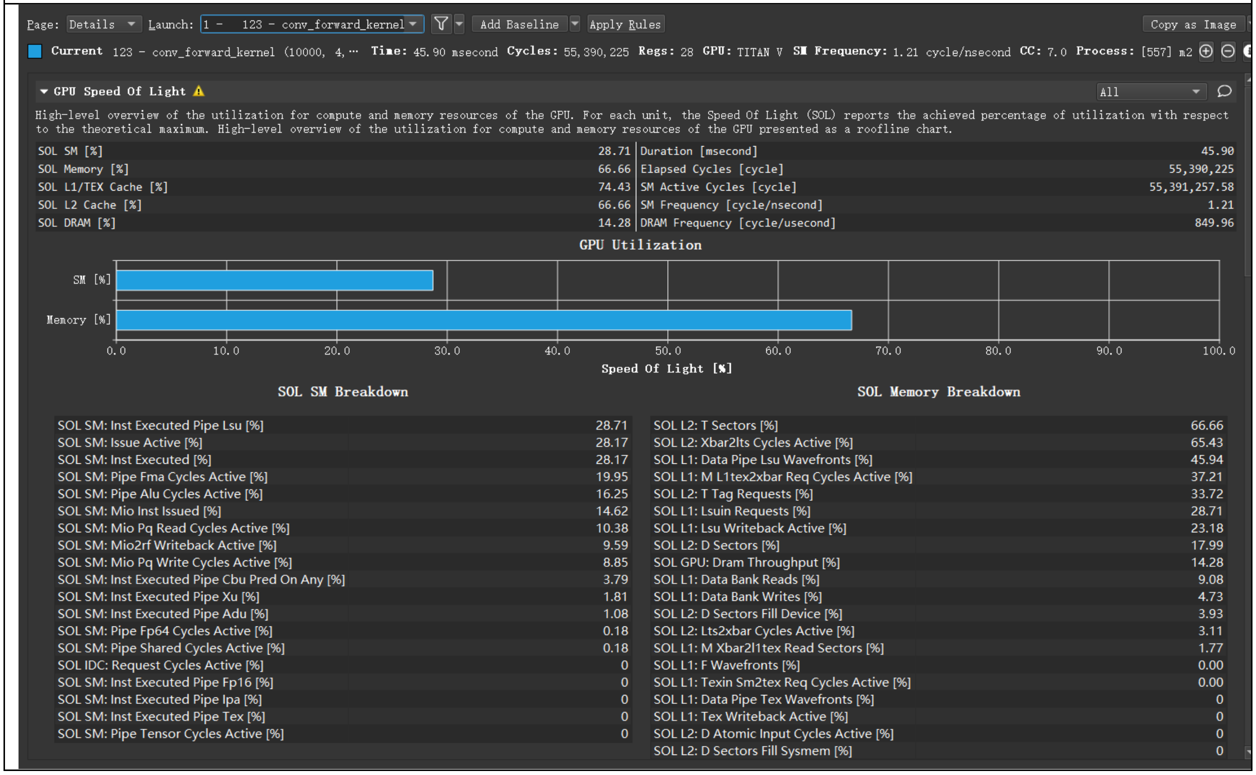
4. List all the CUDA API calls that collectively consumed more than 90% of the API time and what percentage of the API time each call did consume (start with the API call that consumed the most time, then list the next call, until you reach 90% or more).

cudaMemcpy – 75.3% time
cudaDeviceSynchronize – 15.2% time
cudaMalloc – 8.6% time

5. Explain the difference between kernels and CUDA API calls. Please give an example in your explanation for both.

Kernels are device code which can only be invoked by device, while CUDA API calls can be invoked by host code. Therefore, kernels can be designed for many specific tasks, while CUDA API calls play a role of building blocks to call the kernels. For example, cudaMalloc is a basic API call, and conv_forward_kernel in this milestone is the kernel for a specific convolutional layer.

6. Show a screenshot of the GPU SOL utilization



Current 123 - conv_forward_kernel (10000, 4, ··· Time: 45.90 msecond Cycles: 55,390,225 Regs: 28 GPU: TITAN V SM Frequency: 1.21 cycle/nsecond CC: 7.0 Process: [557] m2

GPU Speed Of Light

High-level overview of the utilization for compute and memory resources of the GPU. For each unit, the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum. High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.

SOL SM [%]	28.71	Duration [msecond]	45.90
SOL Memory [%]	66.66	Elapsed Cycles [cycle]	55,390,225
SOL L1/TEX Cache [%]	74.43	SM Active Cycles [cycle]	55,391,257.58
SOL L2 Cache [%]	66.66	SM Frequency [cycle/nsecond]	1.21
SOL DRAM [%]	14.28	DRAM Frequency [cycle/usecond]	849.96

GPU Utilization

SOL SM Breakdown

SOL SM: Inst Executed Pipe Lsu [%]	28.71
SOL SM: Issue Active [%]	28.17
SOL SM: Inst Executed [%]	28.17
SOL SM: Pipe Fma Cycles Active [%]	19.95
SOL SM: Pipe Alu Cycles Active [%]	16.25
SOL SM: Mio Inst Issued [%]	14.62
SOL SM: Mio Pq Read Cycles Active [%]	10.38
SOL SM: Mio2rf Writeback Active [%]	9.59
SOL SM: Mio Pq Write Cycles Active [%]	8.85
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	3.79
SOL SM: Inst Executed Pipe Xu [%]	1.81
SOL SM: Inst Executed Pipe Adu [%]	1.08
SOL SM: Pipe Fp64 Cycles Active [%]	0.18
SOL SM: Pipe Shared Cycles Active [%]	0.18
SOL IDC: Request Cycles Active [%]	0
SOL SM: Inst Executed Pipe Fp16 [%]	0
SOL SM: Inst Executed Pipe Ipa [%]	0
SOL SM: Inst Executed Pipe Tex [%]	0
SOL SM: Pipe Tensor Cycles Active [%]	0

SOL Memory Breakdown

SOL L2: T Sectors [%]	66.66
SOL L2: Xbar2lts Cycles Active [%]	65.43
SOL L1: Data Pipe Lsu Wavefronts [%]	45.94
SOL L1: M L1tex2xbar Req Cycles Active [%]	37.21
SOL L2: T Tag Requests [%]	33.72
SOL L1: Lsuin Requests [%]	28.71
SOL L1: Lsu Writeback Active [%]	23.18
SOL L2: D Sectors [%]	17.99
SOL GPU: Dram Throughput [%]	14.28
SOL L1: Data Bank Reads [%]	9.08
SOL L1: Data Bank Writes [%]	4.73
SOL L2: D Sectors Fill Device [%]	3.93
SOL L2: Lts2xbar Cycles Active [%]	3.11
SOL L1: M Xbar2l1tex Read Sectors [%]	1.77
SOL L1: F Wavefronts [%]	0.00
SOL L1: Texin Sm2tex Req Cycles Active [%]	0.00
SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL L1: Tex Writeback Active [%]	0
SOL L2: D Atomic Input Cycles Active [%]	0
SOL L2: D Sectors Fill Sysmem [%]	0