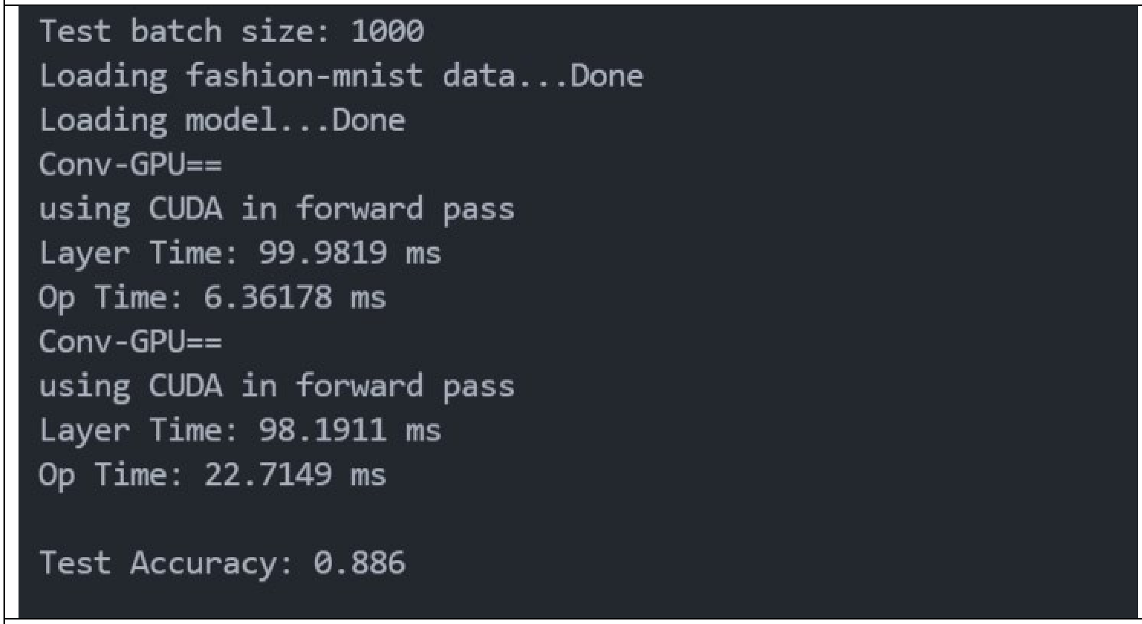
Name: *Zhongbo Zhu*
NetID: *Zhongbo2*
Section: *AL2*

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*").

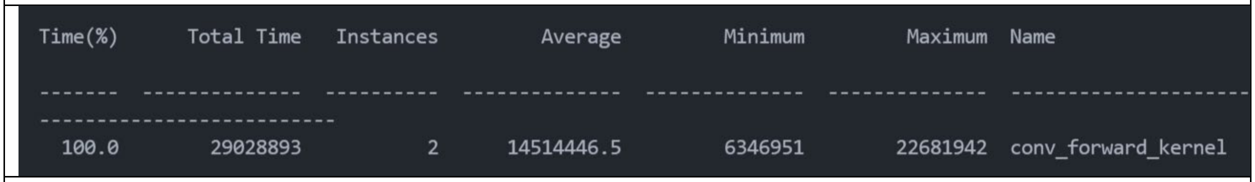


```
Test batch size: 1000
Loading fashion-mnist data...Done
Loading model...Done
Conv-GPU==
using CUDA in forward pass
Layer Time: 99.9819 ms
Op Time: 6.36178 ms
Conv-GPU==
using CUDA in forward pass
Layer Time: 98.1911 ms
Op Time: 22.7149 ms

Test Accuracy: 0.886
```

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

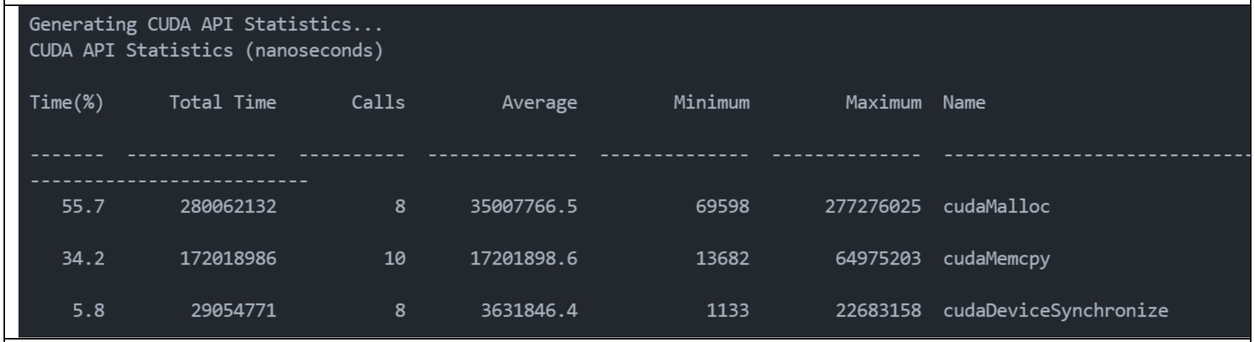
Batch Size	Op Time 1	Op Time 2	Total Execution Time	Accuracy
100	*0.652885 ms*	*2.25154 ms*	*0m5.255s*	*0.86*
1000	*6.36209 ms*	*22.6965 ms*	*0m50.992s*	*0.886*
10000	*63.4182 ms*	*213.689 ms*	*8m33.991s*	*0.8714*

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).



```
 Time(%)      Total Time   Instances         Average         Minimum         Maximum  Name

 -------  -------------  ----------  --------------  --------------  --------------  --------------------
-------------------------
   100.0       29028893           2      14514446.5         6346951        22681942  conv_forward_kernel
```

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).



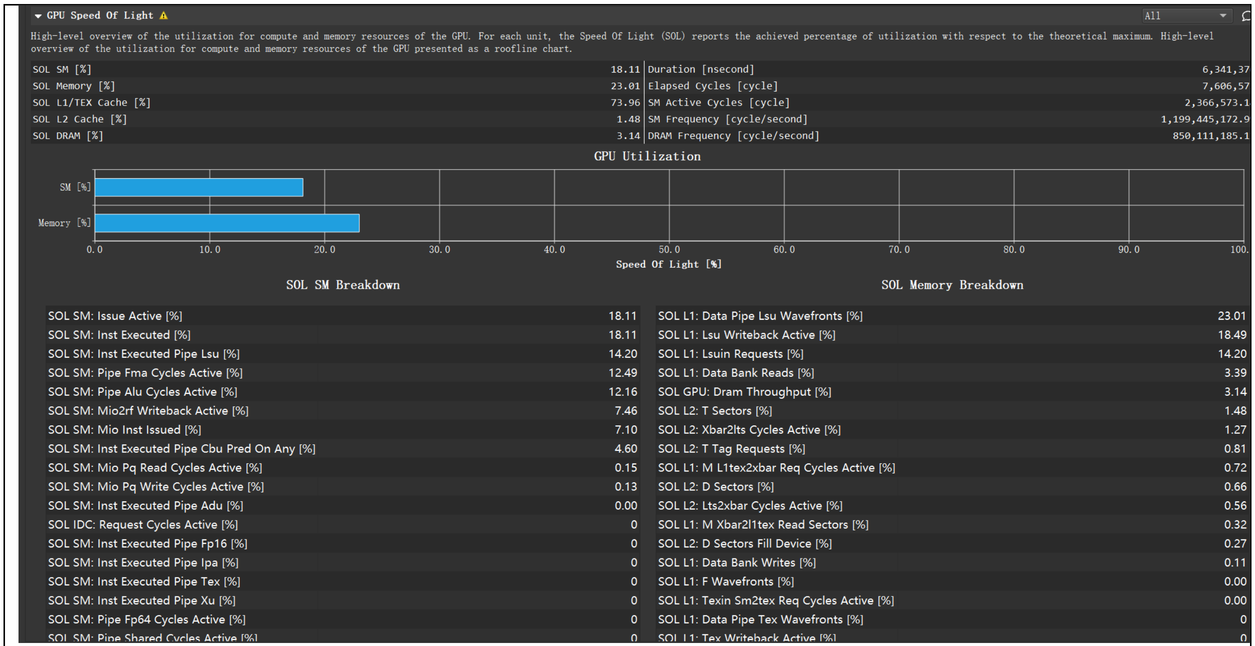
```
Generating CUDA API Statistics...
CUDA API Statistics (nanoseconds)

Time(%)      Total Time       Calls         Average         Minimum         Maximum  Name

-------  -------------  ----------  --------------  --------------  --------------  ----------------------------
-------------------------
   55.7       280062132           8      35007766.5           69598       277276025  cudaMalloc

   34.2       172018986          10      17201898.6           13682        64975203  cudaMemcpy

    5.8        29054771           8       3631846.4            1133        22683158  cudaDeviceSynchronize
```

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

Kernel is the parallel algorithm that is really performed in SM of GPU, so the kernel represents what the GPU cores are executing. The CUDA API is for people to setup the executing tasks of GPU, so that we can deploy tasks on the GPU cores.

6. Show a screenshot of the GPU SOL utilization

For batch 1000

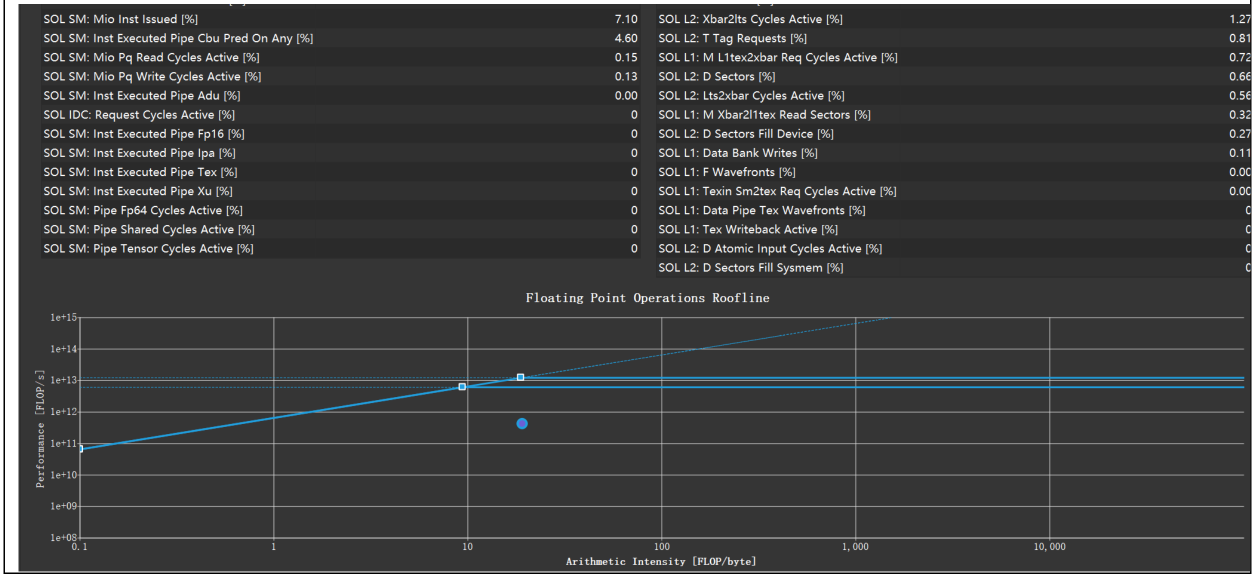


▾ GPU Speed Of Light ⚠

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 [%]	18.11	Duration [nsecond]	6,341,37
SOL Memory [%]	23.01	Elapsed Cycles [cycle]	7,606,57
SOL L1/TEX Cache [%]	73.96	SM Active Cycles [cycle]	2,366,573.1
SOL L2 Cache [%]	1.48	SM Frequency [cycle/second]	1,199,445,172.9
SOL DRAM [%]	3.14	DRAM Frequency [cycle/second]	850,111,185.1

SOL SM Breakdown		SOL Memory Breakdown	
SOL SM: Issue Active [%]	18.11	SOL L1: Data Pipe Lsu Wavefronts [%]	23.01
SOL SM: Inst Executed [%]	18.11	SOL L1: Lsu Writeback Active [%]	18.49
SOL SM: Inst Executed Pipe Lsu [%]	14.20	SOL L1: Lsuin Requests [%]	14.20
SOL SM: Pipe Fma Cycles Active [%]	12.49	SOL L1: Data Bank Reads [%]	3.39
SOL SM: Pipe Alu Cycles Active [%]	12.16	SOL GPU: Dram Throughput [%]	3.14
SOL SM: Mio2rf Writeback Active [%]	7.46	SOL L2: T Sectors [%]	1.48
SOL SM: Mio Inst Issued [%]	7.10	SOL L2: Xbar2lts Cycles Active [%]	1.27
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	4.60	SOL L2: T Tag Requests [%]	0.81
SOL SM: Mio Pq Read Cycles Active [%]	0.15	SOL L1: M L1tex2xbar Req Cycles Active [%]	0.72
SOL SM: Mio Pq Write Cycles Active [%]	0.13	SOL L2: D Sectors [%]	0.66
SOL SM: Inst Executed Pipe Adu [%]	0.00	SOL L2: Lts2xbar Cycles Active [%]	0.56
SOL IDC: Request Cycles Active [%]	0	SOL L1: M Xbar2l1tex Read Sectors [%]	0.32
SOL SM: Inst Executed Pipe Fp16 [%]	0	SOL L2: D Sectors Fill Device [%]	0.27
SOL SM: Inst Executed Pipe Ipa [%]	0	SOL L1: Data Bank Writes [%]	0.11
SOL SM: Inst Executed Pipe Tex [%]	0	SOL L1: F Wavefronts [%]	0.00
SOL SM: Inst Executed Pipe Xu [%]	0	SOL L1: Texin Sm2tex Req Cycles Active [%]	0.00
SOL SM: Pipe Fp64 Cycles Active [%]	0	SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL SM: Pipe Shared Cycles Active [%]	0	SOL L1: Tex Writeback Active [%]	0



SOL SM: Mio Inst Issued [%]	7.10	SOL L2: Xbar2lts Cycles Active [%]	1.27
SOL SM: Inst Executed Pipe Cbu Pred On Any [%]	4.60	SOL L2: T Tag Requests [%]	0.81
SOL SM: Mio Pq Read Cycles Active [%]	0.15	SOL L1: M L1tex2xbar Req Cycles Active [%]	0.72
SOL SM: Mio Pq Write Cycles Active [%]	0.13	SOL L2: D Sectors [%]	0.66
SOL SM: Inst Executed Pipe Adu [%]	0.00	SOL L2: Lts2xbar Cycles Active [%]	0.56
SOL IDC: Request Cycles Active [%]	0	SOL L1: M Xbar2l1tex Read Sectors [%]	0.32
SOL SM: Inst Executed Pipe Fp16 [%]	0	SOL L2: D Sectors Fill Device [%]	0.27
SOL SM: Inst Executed Pipe Ipa [%]	0	SOL L1: Data Bank Writes [%]	0.11
SOL SM: Inst Executed Pipe Tex [%]	0	SOL L1: F Wavefronts [%]	0.00
SOL SM: Inst Executed Pipe Xu [%]	0	SOL L1: Texin Sm2tex Req Cycles Active [%]	0.00
SOL SM: Pipe Fp64 Cycles Active [%]	0	SOL L1: Data Pipe Tex Wavefronts [%]	0
SOL SM: Pipe Shared Cycles Active [%]	0	SOL L1: Tex Writeback Active [%]	0

SOL SM: Pipe Tensor Cycles Active [%]	0	SOL L2: D Atomic Input Cycles Active [%]	0
		SOL L2: D Sectors Fill Sysmem [%]	0