**ECE 408 Project Report**

**Team Name:** fast\_af

**Names:** Omar Taha, Jason Gong, Shreyas Byndoor

**NetIDs:** otaha2, xg8, byndoor2

**Rai IDs:** 5c78c4b084318364bab96f01,

**Affiliation**: Chicago City Scholars

**Milestone 1**

Kernels that collectively consumed more than 90% of the program time:

* [CUDA memcpy HtoD]
* cudnn::detail::implicit\_convolve\_sgemm
* volta\_cgemm\_64x32\_tn
* op\_generic\_tensor\_kernel
* fft2d\_c2r\_32x32
* volta\_sgemm\_128x128\_tn
* cudnn::detail::pooling\_fw\_4d\_kernel
* fft2d\_r2c\_32x32

CUDA API calls that collectively consume more than 90% of the program time:

* cudaStreamCreateWithFlags
* cudaMemGetInfo
* cudaFree

Kernels vs. API Calls:

CUDA Kernels are simply defined as regular C functions. However, unlike typical C functions, CUDA Kernels are executed N times in parallel by N different CUDA threads. Meanwhile, CUDA APIs provide C functions that execute on the host to allocate and deallocate device memory, transfer data between host memory and device memory, manage systems with multiple devices, etc. They are not executed by each CUDA thread as a Kernel is.

Output of rai running MXNet on the CPU:

Loading fashion-mnist data... done

Loading model... done

New Inference

EvalMetric: {'accuracy': 0.8236}

9.30user 3.46system 0:05.34elapsed 239%CPU (0avgtext+0avgdata 2471560maxresident)k

0inputs+2824outputs (0major+666761minor)pagefaults 0swaps

Program run time:

0:05.34 elapsed

Output of rai running MXNet on the GPU:

Loading fashion-mnist data... done

Loading model... done

New Inference

EvalMetric: {'accuracy': 0.8236}

4.41user 3.28system 0:04.34elapsed 177%CPU (0avgtext+0avgdata 2837968maxresident)k

0inputs+4552outputs (0major+661333minor)pagefaults 0swaps

Program run time:

0:04.34 elapsed

**Milestone 2**

program execution time:

0:15.52 elapsed

Op Times:

1) Op Time: 2.826305

2) Op Time: 11.143122

**Milestone 3**

Correctness & Timing:

|  |  |  |  |
| --- | --- | --- | --- |
|  | DataSet 100 | DataSet 1000 | DataSet 10000 |
| Op Time #1 | 0.000079 | 0.000558 | 0.005636 |
| Op Time #2 | 0.000216 | 0.001974 | 0.021531 |
| Correctness | 0.84 | 0.852 | 0.8397 |

nvprof profile:

A screenshot of a social media post

Description automatically generated

**Milestone 4**

Base Profiling:

**A screenshot of a social media post

Description automatically generated**

Optimization #1: Weight Matrix in Constant Memory

By copying the weight matrix into constant memory, we are drastically decreasing the number of global memory reads to 1 per weight data point. The kernel then reads from constant memory for the weight matrix. **Using NVVP, we found that the average kernel duration decreased from 12.854 ms to 12.144 ms.** Below is the nvprof profiling of the kernel + Weight in Constant Memory Optimization.

A screenshot of a cell phone

Description automatically generated