

UNIVERSITY OF ILLINOIS AT URBANA CHAMPAIGN

APPLIED PARALLEL PROGRAMMING

Team: wandering-gpu

---

## Final Project

---

*Author*

Alvin Sun

Yuqi Xue

Yan Miao

*Net ID*

yixiaos3

yuqixue2

yanmiao2

March 12, 2019

# Milestone 1

## 1 Kernel Statistics

Time(%)	Time	Calls	Avg	Min	Max	Name
40.10%	16.788ms	20	839.42us	1.1200us	16.155ms	[CUDA memcpy HtoD]
20.18%	8.4497ms	1	8.4497ms	8.4497ms	8.4497ms	void cudnn::detail::implicit_convolve_sgemm
11.81%	4.9434ms	1	4.9434ms	4.9434ms	4.9434ms	volta_cgemm_64x32_tn
7.05%	2.9497ms	2	1.4748ms	25.568us	2.9241ms	void op_generic_tensor_kernel
5.69%	2.3830ms	1	2.3830ms	2.3830ms	2.3830ms	void fft2d_c2r_32x32
5.59%	2.3404ms	1	2.3404ms	2.3404ms	2.3404ms	volta_sgemm_128x128_tn
4.55%	1.9059ms	1	1.9059ms	1.9059ms	1.9059ms	void cudnn::detail::pooling_fw_4d_kernel
4.18%	1.7480ms	1	1.7480ms	1.7480ms	1.7480ms	void fft2d_r2c_32x32

## 2 CUDA API Statistics

Time(%)	Time	Calls	Avg	Min	Max	Name
41.94%	2.94373s	22	133.81ms	13.721us	1.52482s	cudaStreamCreateWithFlags
34.43%	2.41664s	24	100.69ms	97.853us	2.41057s	cudaMemGetInfo
20.93%	1.46898s	19	77.315ms	817ns	393.98ms	cudaFree

## 3 Differences Between Kernels & API Calls

A CUDA kernel is an extended C function that, when called, are executed multiple times in parallel by different CUDA threads on the GPU. The CUDA APIs are programming interfaces that allow the programmer to use the CUDA device, i.e. the GPU. Kernel functions are written by the programmer and are meant to execute specific (mostly computation-intensive) tasks on the GPU, while API calls are provided by the CUDA library and are to manage the CUDA runtime environment and mostly prepare for the execution of kernels. While kernels are always executed by CUDA cores, CUDA APIs do not necessarily involve the execution of CUDA cores.

## 4 MXNet CPU Execution

```

0 Loading fashion-mnist data... done
1 Loading model... done
2 New Inference
3 EvalMetric: {'accuracy': 0.8236}

```

**Run Time.** 5.06s

## 5 MXNet GPU Execution

```

0 Loading fashion-mnist data... done
1 Loading model... done
2 New Inference
3 EvalMetric: {'accuracy': 0.8236}

```

**Run Time:** 4.40s

## Milestone 2

Full CPU Time	11.32s	Full CPU Time	0.250576s	Full CPU Time	1.08s
First Layer Time	2.405296s	First Layer Time	0.756567s	First Layer Time	0.035050s
Second Layer Time	7.342860	Second Layer Time	2.03s	Second Layer Time	0.075312s
10000 images		1000 images		100 images	

Table 1: CPU Run Time Statistics