

# ECE 408 Final Project:

Member1: Yuxiang Li (yl48)

Member2: Tianyu Sun (tianyus2)

Member3: Chi Wang Ng (cng10)

## Milestone 1:

**Report: Include a list of all kernels that collectively consume more than 90% of the program time.**

1. [CUDA memcpy HtoD]
2. Volta\_scudnn\_128x32\_relu\_interior\_nn\_v1
3. void cudnn::detail::implicit\_convolve\_sgemm<float, float, int=1024, int=5, int=5, int=3, int=3, int=3, int=1, bool=1, bool=0, bool=1>(int, int, int, float const \*, int, float\*, cudnn::detail::implicit\_convolve\_sgemm<float, float, int=1024, int=5, int=5, int=3, int=3, int=3, int=1, bool=1, bool=0, bool=1>\*, kernel\_conv\_params, int, float, float, int, float, float, int, int)
4. void cudnn::detail::activation\_fw\_4d\_kernel<float, float, int=128, int=1, int=4, cudnn::detail::tanh\_func<float>>(cudnnTensorStruct, float const \*, cudnn::detail::activation\_fw\_4d\_kernel<float, float, int=128, int=1, int=4, cudnn::detail::tanh\_func<float>>, cudnnTensorStruct\*, float, cudnnTensorStruct\*, int, cudnnTensorStruct\*)
5. Volta\_sgemm\_128x128\_tn
6. void cudnn::detail::pooling\_fw\_4d\_kernel<float, float, cudnn::detail::maxpooling\_func<float, cudnnNanPropagation\_t=0>, int=0, bool=0>(cudnnTensorStruct, float const \*, cudnn::detail::pooling\_fw\_4d\_kernel<float, float, cudnn::detail::maxpooling\_func<float, cudnnNanPropagation\_t=0>, int=0, bool=0>, cudnnTensorStruct\*, cudnnPoolingStruct, float, cudnnPoolingStruct, int, cudnn::reduced\_divisor, float)
7. void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::ScalarExp<float>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2, int)
8. void mshadow::cuda::SoftmaxKernel<int=8, float, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>>(mshadow::gpu, int=2, unsigned int)
9. void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::ScalarExp<float>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)

10. volta\_sgemm\_32x32\_sliced1x4\_tn

**Report: Include a list of all CUDA API calls that collectively consume more than 90% of the program time.**

1. cudaStreamCreateWithFlags
2. cudaMemGetInfo
3. cudaFree
4. cudaFuncSetAttribute
5. cudaMemcpy2DAsync
6. cudaStreamSynchronize
7. cudaEventCreateWithFlags
8. cudaMalloc
9. cudaGetDeviceProperties
10. cudaMemcpy

**Report: Include an explanation of the difference between kernels and API calls**

Kernels are executed on the device in parallel by different CUDA threads.

Kernels are built on top of a lower-level API calls, which is also accessible by the application. Kernels provides an additional level of control by exposing lower-level concepts such as CUDA contexts - the analogue of host processes for the device - and CUDA modules - the analogue of dynamically loaded libraries for the device.

**Report: Show output of rai running MXNet on the CPU**

```
Loading fashion-mnist data... done
Loading model... done
New Inference
EvalMetric: {'accuracy': 0.8177}
19.33user 3.85system 0:13.18elapsed 175%CPU (0avgtext+0avgdata
5956088maxresident)k
0inputs+2856outputs (0major+1584783minor)pagefaults
0swaps
```

**Report: List program run time**

13.18s

**Report: Show output of rai running MXNet on the GPU**

```
Loading fashion-mnist data... done
Loading model... done
New Inference
EvalMetric: {'accuracy': 0.8177}
```

```
4.49user 2.54system 0:05.01elapsed 140%CPU (0avgtext+0avgdata  
2840048maxresident)k  
0inputs+4568outputs (0major+704659minor)pagefaults 0swaps
```

**Report: List program run time**

5.01s

## Milestone 2:

Data size: 10000

```
* Running /usr/bin/time python m2.1.py
Loading fashion-mnist data... done
Loading model...[22:24:03] src/nnvm/legacy_json_util.cc:204: Warning: loading symbol
00. May cause undefined behavior. Please update MXNet if you encounter any issue
done
New Inference
Op Time: 25.142652
Op Time: 151.186232
Correctness: 0.8171 Model: ece408
187.16user 7.83system 2:59.77elapsed 108%CPU (0avgtext+0avgdata 5866708maxresident)k
0inputs+0outputs (0major+2250609minor)pagefaults 0swaps
```

### Milestone 3:

Data size: 100

```
New Inference
Values: 66,66,5,5,25
Op Time: 0.000477
Values: 27,27,2,2,4
Op Time: 0.001654
Correctness: 0.85 Model: ece408
4.12user 2.61system 0:04.38elapsed 153%CPU (0avgtext+0avgdata 2641584maxresident)k
```

Data size: 1000

```
* Running /usr/bin/time python m3.1.py 1000
Loading fashion-mnist data... done
Loading model... done
New Inference
Values: 66,66,5,5,25
Op Time: 0.004381
Values: 27,27,2,2,4
Op Time: 0.016349
Correctness: 0.827 Model: ece408
3.96user 2.33system 0:04.10elapsed 153%CPU (0avgtext+0avgdata 2644080maxresident)k
```

Data size : 10000

```
* Running /usr/bin/time python m3.1.py
Loading fashion-mnist data... done
Loading model... done
New Inference
Values: 66,66,5,5,25
Op Time: 0.043434
Values: 27,27,2,2,4
Op Time: 0.152497
Correctness: 0.8171 Model: ece408
4.13user 2.63system 0:04.48elapsed 150%CPU (0avgtext+0avgdata 2820616maxresident)k
```

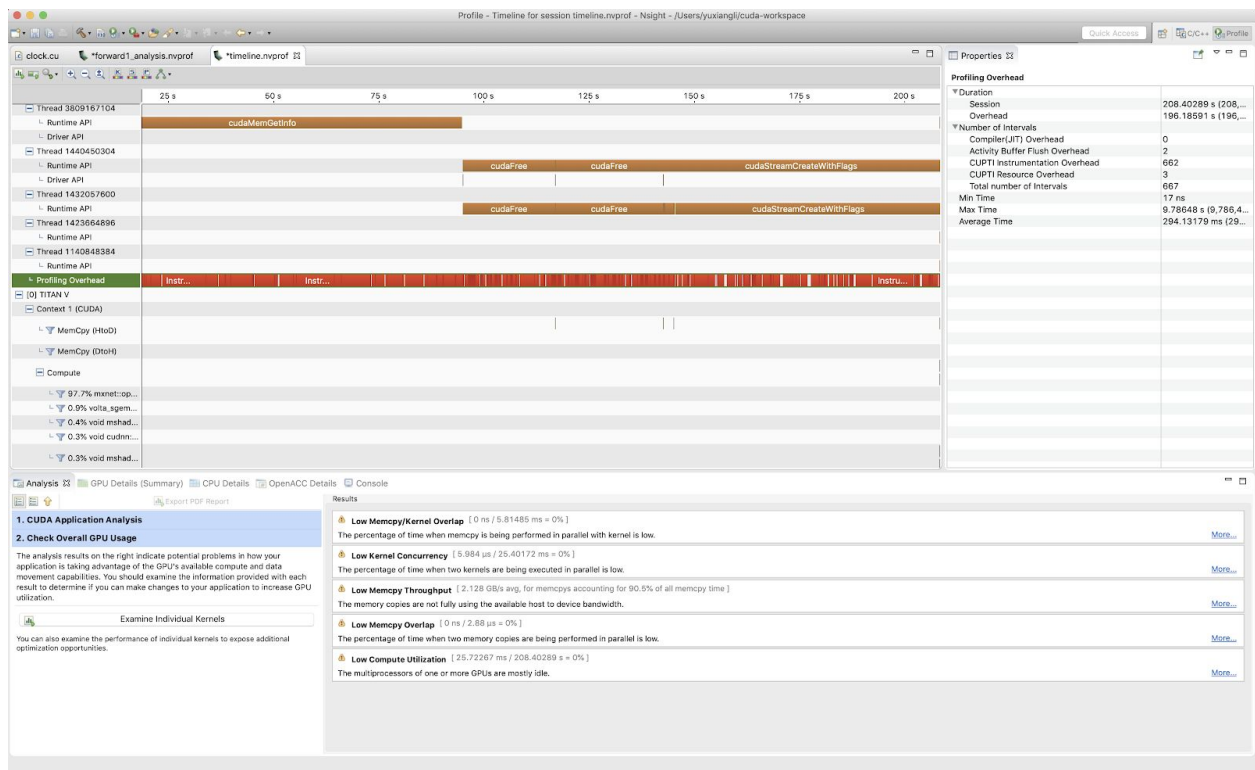
## Profile:

```
* Running nvprof python m3.1.py
Loading fashion-mnist data... done
==474== NVPROF is profiling process 474, command: python m3.1.py
Loading model... done
New Inference
Values: 66,66,5,5,25
Op Time: 0.039462
Values: 27,27,2,2,4
Op Time: 0.146472
Correctness: 0.8171 Model: ece408
==474== Profiling application: python m3.1.py
==474== Profiling result:
   Type  Time(%)   Time     Calls   Avg       Min       Max    Name
GPU activities: 73.24% 185.81ms      2 92.906ms 39.394ms 146.42ms void mxnet::op::forward_kernel<mshadow::gpu, float>(float*, mxnet::op::forward_kernel<mshadow::gpu, float> const *, int, int, int, int, int, int)
   13.54% 34.348ms     20 1.7174ms 1.0240us 32.382ms [CUDA memcpy HtoD]
   5.66% 14.370ms      2 7.1850ms 2.5314ms 11.839ms void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, float>, float, int=1>, float>>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
   2.88% 7.2949ms      2 3.6475ms 24.383us 7.2706ms void cudnn::detail::activation_fw_4d_kernel<float, float, int=128, int=1, cudnn::detail::tanh_func<float>>(cudnnTensorStruct, float const *, cudnn::detail::activation_fw_4d_kernel<float, float, int=128, int=1, cudnn::detail::tanh_func<float>>, cudnnTensorStruct*, float, cudnnTensorStruct*, int, cudnnTensorStruct*)
   2.69% 6.8249ms      1 6.8249ms 6.8249ms 6.8249ms volta_sgemm_128x128_tn
   1.73% 4.3814ms      1 4.3814ms 4.3814ms void cudnn::detail::pooling_fw_4d_kernel<float, float, cudnn::detail::maxpooling_func<float, cudnnNanPropagation_t=0>, int=0, bool=0>(cudnnTensorStruct, float const *, cudnn::detail::pooling_fw_4d_kernel<float, float, cudnn::detail::maxpooling_func<float, cudnnNanPropagation_t=0>, int=0, bool=0>, cudnnTensorStruct*, float, cudnnPoolingStruct, int, cudnn::reduced_divisor, float)
   0.18% 457.79us      1 457.79us 457.79us 457.79us void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, float>, float>, mshadow::expr::Plan<mshadow::expr::ScalarExp<float>, float>>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2, int)
```

```
   0.03% 68.384us      1 68.384us 68.384us 68.384us void mshadow::cuda::SoftmaxKernel<int=8, float, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, float>, float>, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, float>, float>>>(mshadow::gpu, int=2, unsigned int)
   0.02% 53.471us     13 4.1130us 1.0560us 19.488us void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, float>, float>, mshadow::expr::Plan<mshadow::expr::ScalarExp<float>, float>>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
   0.01% 30.720us      1 30.720us 30.720us 30.720us volta_sgemm_32x32_sliced1x4_tn
   0.01% 26.176us      2 13.088us 2.3360us 23.840us void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, float>, float>, mshadow::expr::Plan<mshadow::expr::Broadcast10Exp<mshadow::Tensor<mshadow::gpu, float>, float, int=2, int=1>, float>>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
   0.00% 7.8720us      8 984ns 928ns 1.3440us [CUDA memset]
   0.00% 5.5680us      1 5.5680us 5.5680us 5.5680us [CUDA memcpy DtoH]
   0.00% 4.6080us      1 4.6080us 4.6080us 4.6080us void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, float>, float>, mshadow::expr::Plan<mshadow::expr::ReduceWithAxisExp<mshadow::red::maximum, mshadow::Tensor<mshadow::gpu, float>, float, int=3, bool=1, int=2>, float>>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
API calls: 39.06% 2.7336s      22 12
4.26ms 13.371us 1.4132s cudaStreamCreateWithFlags
33.95% 2.3755s      22 107.98ms 70.453us 2.3708s cudaMemGetInfo
21.18% 1.4819s      18 82.331ms 319ns 389.16ms cudaFree
2.86% 200.22ms      6 33.371ms 3.6600us 146.44ms cudaDeviceSynchronize
1.00% 69.666ms      9 7.7406ms 28.882us 32.581ms cudaMemcpy2DAsync
0.79% 55.338ms      384 144.11us 277ns 24.881ms cudaFuncSetAttribute
0.57% 39.644ms      66 600.66us 5.3850us 18.116ms cudaMalloc
0.26% 18.504ms      29 638.06us 4.0210us 11.033ms cudaStreamSynchronize
0.17% 12.174ms      216 56.362us 321ns 10.997ms cudaEventCreateWithFlags
0.07% 4.8975ms      4 1.2244ms 420.15us 1.8425ms cudaGetDeviceProperties
0.03% 2.3046ms      375 6.1450us 100ns 330.44us cuDeviceGetAttribute
0.01% 723.54us      2 361.77us 47.544us 676.00us cudaHostAlloc
0.01% 702.04us      4 175.51us 97.339us 276.17us cuDeviceTotalMem
0.01% 617.16us      27 22.857us 6.6540us 61.822us cudaLaunchKernel
0.01% 521.69us      12 43.474us 5.8350us 161.53us cudaMemcpy
```

```
0.01% 504.89us      4 126.22us 67.027us 218.92us cudaStreamCreate
0.00% 268.88us      4 67.219us 44.463us 105.68us cuDeviceGetName
0.00% 246.90us      8 30.862us 8.3700us 106.66us cudaMemsetAsync
0.00% 156.17us      8 19.521us 13.190us 41.745us cudaStreamCreateWithPriority
0.00% 131.99us      202 653ns 198ns 16.352us cudaDeviceGetAttribute
0.00% 104.08us      29 3.5890us 515ns 11.512us cudaSetDevice
0.00% 25.255us      18 1.4030us 238ns 2.9150us cudaGetDevice
0.00% 22.131us      6 3.6880us 746ns 7.9230us cudaEventCreate
0.00% 9.0100us      2 4.5050us 3.5650us 5.4450us cudaEventRecord
0.00% 7.5560us      20 377ns 143ns 771ns cudaPeekAtLastError
0.00% 4.3460us      2 2.1730us 1.6430us 2.7030us cudaHostGetDevicePointer
0.00% 3.8870us      1 3.8870us 3.8870us 3.8870us cuDeviceGetPCIBusId
0.00% 3.7330us      2 1.8660us 1.4170us 2.3160us cudaDeviceGetStreamPriorityRange
0.00% 3.5260us      4 881ns 460ns 1.7870us cudaGetDeviceCount
0.00% 2.9070us      6 484ns 145ns 1.2550us cuDeviceGetCount
0.00% 2.2660us      5 453ns 206ns 876ns cuDeviceGet
0.00% 2.2170us      3 739ns 470ns 1.2650us cuInit
0.00% 1.7290us      3 576ns 291ns 1.1250us cuDriverGetVersion
0.00% 1.4320us      5 286ns 165ns 461ns cudaGetLastError
```

## Analysis:



The performance is slow because we are reading from and writing to the global memory, which has a lower throughput. The performance is also limited because we did not reuse some of the elements and reading them from global memory again and again. For example, the filter mask never changes but we read them from the global memory every time we need them. Thus we spent more time than necessary accessing these elements and our performance is limited.