ECE 408 Report

Team Name: tiledtitans

Members:

- Jackson DeDobbelaere (dedobbe2)
- Matthew Grossfeld (grossfe2)
- Xinbo Wu (xinbowu2)

All on campus students

MILESTONE 1

List of all kernels that collectively consume more than 90% of the program time:

- [CUDA memcpy HtoD]
- 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)
- Volta_cgemm_64x32_tn
- void op_generic_tensor_kernel<int=2, float, float, float, int=256, cudnnGenericOp_t=7, cudnnNanPropagation_t=0, cudnnDimOrder_t=0, int=1>(cudnnTensorStruct, float*, cudnnTensorStruct, float const *, cudnnTensorStruct, float const *, float, float, float, float, dimArray, reducedDivisorArray)
- Volta_sgemm_128x128_tn
- void fft2d_c2r_32x32<float, bool=0, bool=0, unsigned int=1, bool=0, bool=0>(float*, float2 const *, int, int, int, int, int, int, int, int, int, float, float, cudnn::reduced_divisor, bool, float*, float*, int2, int, int)
- 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)

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

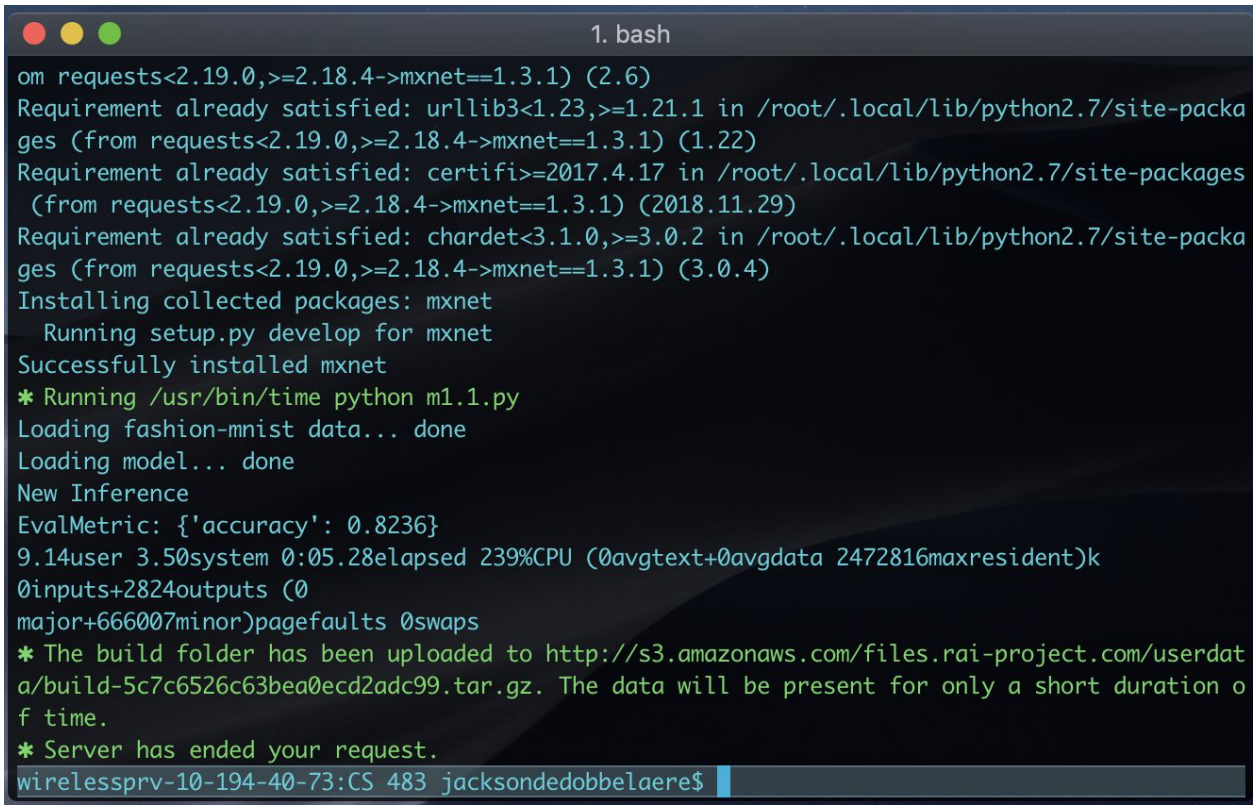
- cudaStreamCreateWithFlags
- cudaMemGetInfo
- cudaFree

Explanation of the difference between kernels and API calls:

API calls are done with the CPU and execute from the time the call is made to the time it is returned. Kernels are ran on the GPU and are measured by the total time that the kernel is executing and running instructions. cudaLaunchKernel is an API function called from the CPU

that can launch kernels from the GPU. The profiling done on the API accounts for all of the launch overhead while the kernel timing only contains a small portion of it.

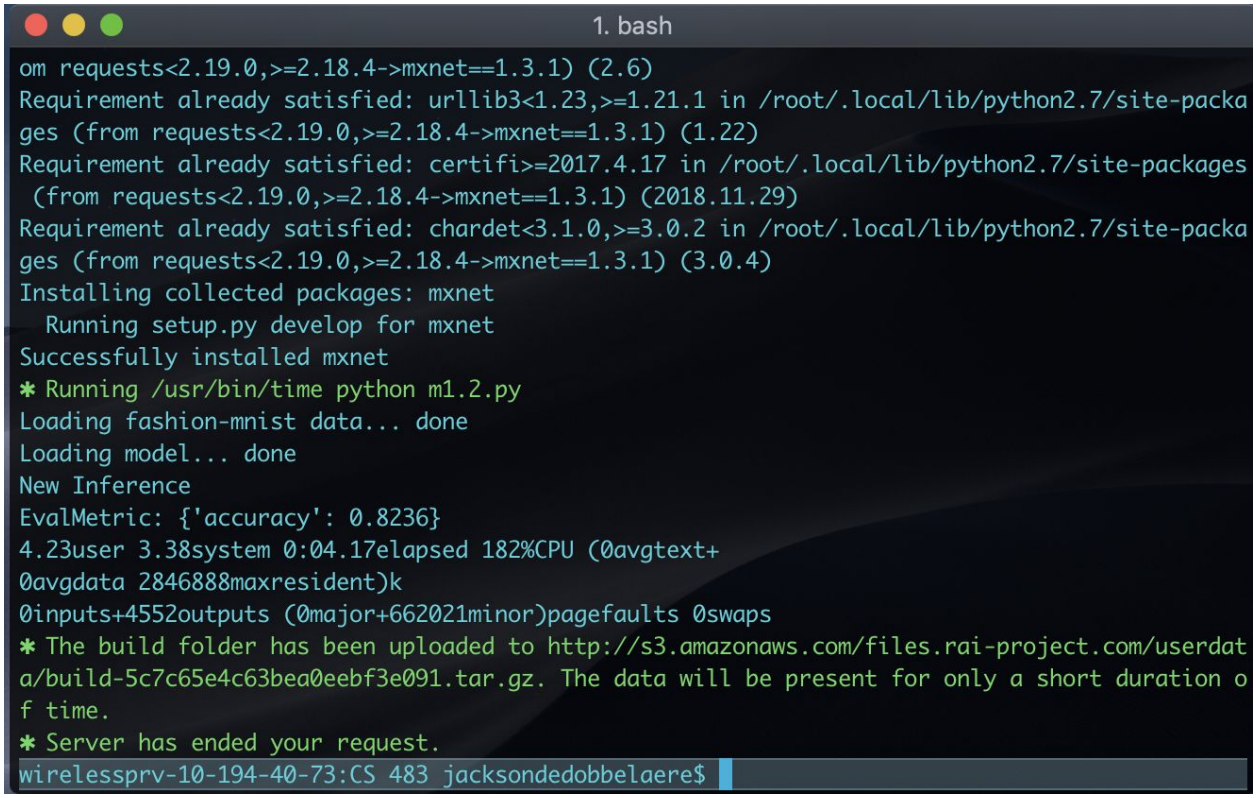
Output of rai running MXNet on the CPU:



Program run time of MXNet on the CPU:

- User: 9.14
- System: 3.50
- Elapsed: 0:05.28

Output of rai running MXNet on the GPU:



Program run time of MXNet on the GPU:

- User: 4.23
- System: 3.38
- Elapsed: 0:04.17

MILESTONE 2

100 Images:

- **Whole Program Execution Time:**
  - User: 2.77
  - System: 2.65
  - Elapsed: 0:01.03
- **Layer 1 Op Time:** 0.034247
- **Layer 2 Op Time:** 0.074316

1,000 Images:

- **Whole Program Execution Time:**
  - User: 4.24
  - System: 3.15
  - Elapsed: 0:01.95
- **Layer 1 Op Time:** 0.238204
- **Layer 2 Op Time:** 0.743339

10,000 Images:

- **Whole Program Execution Time:**
  - User: 14.97
  - System: 4.47
  - Elapsed: 0:11.37
- **Layer 1 Op Time:** 2.427247
- **Layer 2 Op Time:** 7.383249

MILESTONE 3

100 Images:

- **Correctness:** 0.84
- **Whole Program Execution Time:**
    - User: 4.36
    - System: 3.27
    - Elapsed: 0:04.34
- **Layer 1 Op Time:** 0.000102
- **Layer 2 Op Time:** 0.000236

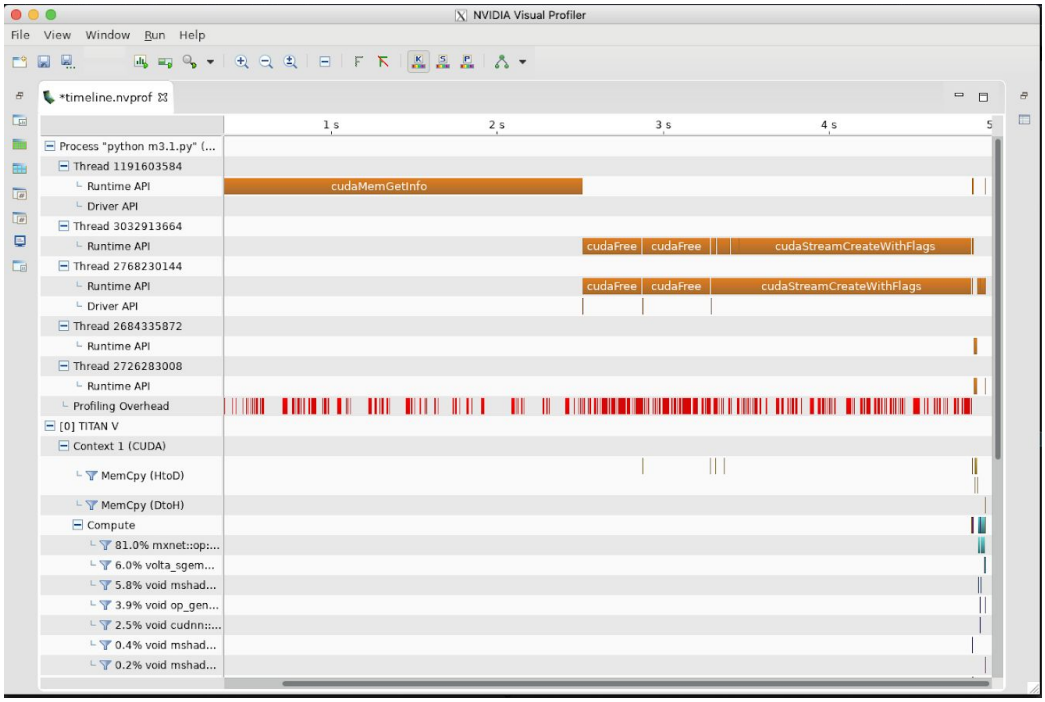
1,000 Images:

- **Correctness:** 0.852
- **Whole Program Execution Time:**
    - User: 4.42
    - System: 3.39
    - Elapsed: 0:04.22
- **Layer 1 Op Time:** 0.000908
- **Layer 2 Op Time:** 0.002467

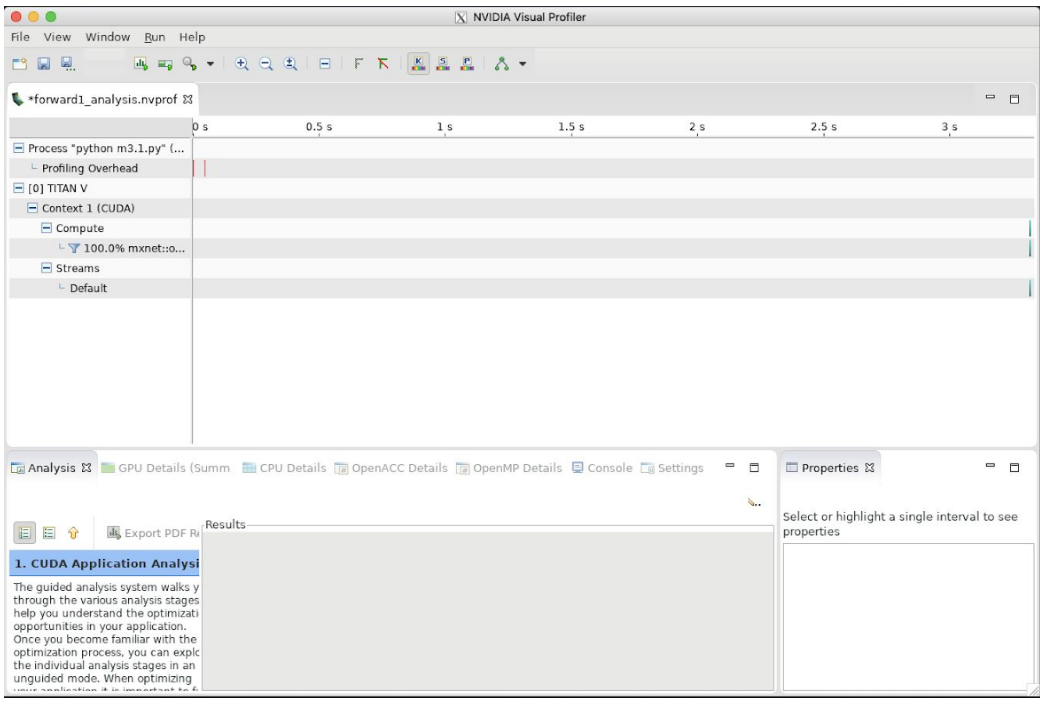
10,000 Images:

- **Correctness:** 0.8397
- **Whole Program Execution Time:**
    - User: 4.33
    - System: 3.31
    - Elapsed: 0:04.34
- **Layer 1 Op Time:** 0.009238
- **Layer 2 Op Time:** 0.024392

NVVP output of timeline.nvprof:



NVVP output of forward1_analysis.nvprof:



NVVP output of forward2_analysis.nvprof: