ECE 408
Spring 2019

Final Project Lab Report

Team: sphavwv
Xinyi Guo (xinyig2, City Scholars)
Kevin So (kevinso2, On Campus)
Jiamin Zhu (jiaminz2, City Scholars)

MILESTONE 1

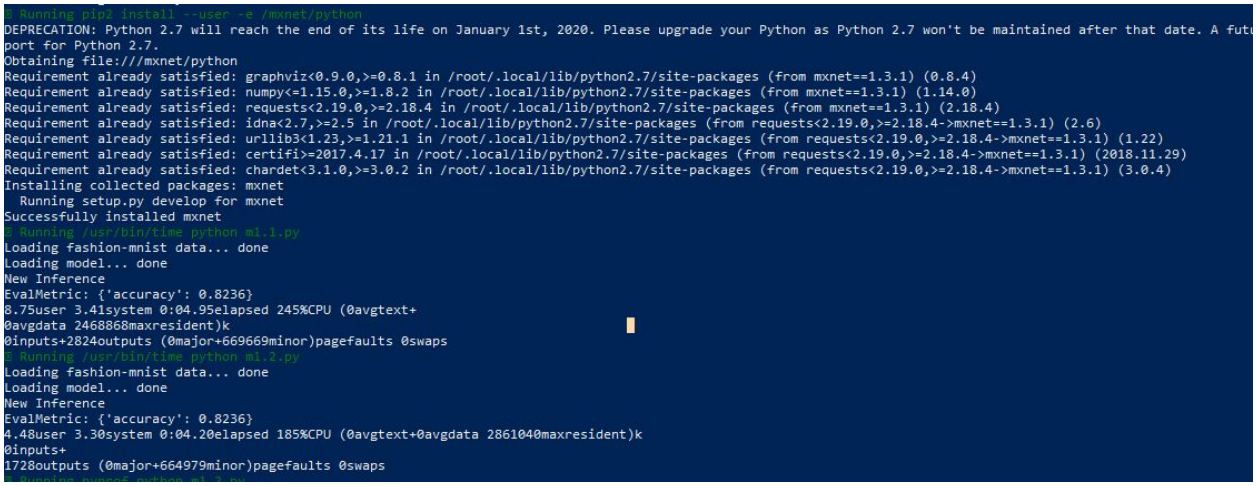
1. Include a list of all kernels that collectively consume more than 90% of the program time.
   - [CUDA memcpy HtoD]
   - void cudnn::detail::implicit_convolve_sgemm...
   - volta_cgemm_64x32_tn
   - void op_generic_tensor_kernel
   - Volta_sgemm_128x128_tn
   - void fft2d_c2r_32x32

2. Include a list of all CUDA API calls that collectively consume more than 90% of the program time.
   - cudaStreamCreateWithFlags
   - cudaMemGetInfo
   - cudaFree

3. Include an explanation of the difference between kernels and API calls.

One of the highest percentages from API calls came from cudaStreamCreateWithFlags which is a __host__ __device__ function. API calls seem to deal with things like syncthreads and organizing the GPU's execution while the GPU deals with the actual computing tasks.Regarding the time percentages for the GPU, we see that the memory copy from host to device is taking up the most due to global memory read and write cycles and the fact that it is called 20 times.

4. Show output of rai running MXNet on the GPU & CPU

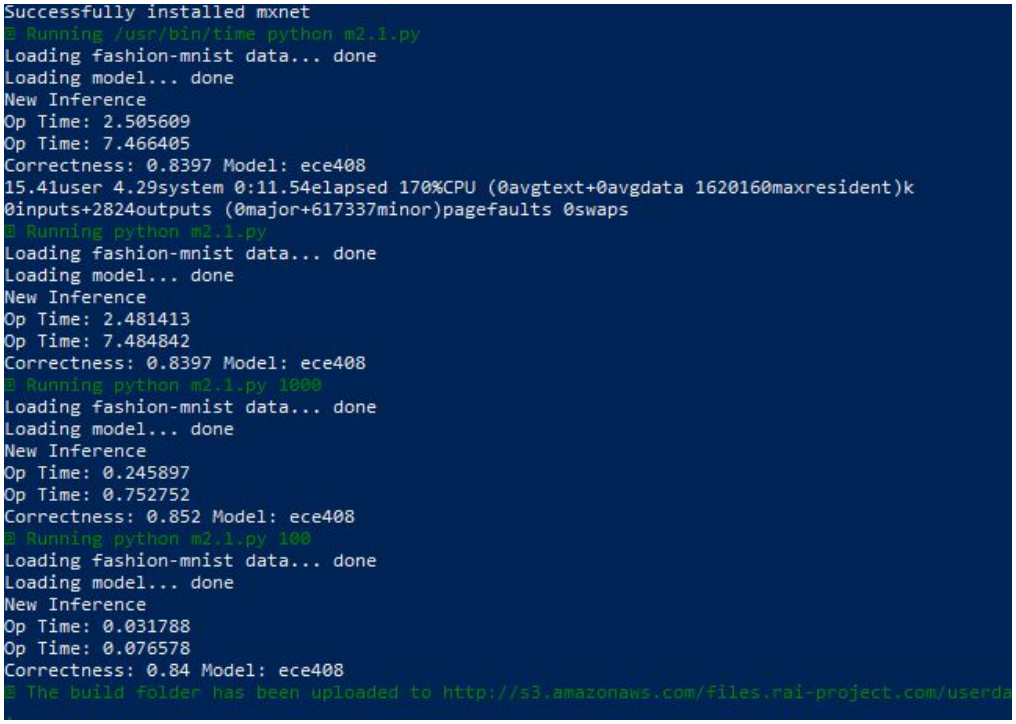


CPU: List program run time

8.75cpu seconds (user mode) 3.41 cpu seconds system (kernel mode) 4.95s elapsed real time

GPU: List program run time
4.48 cpu seconds (user mode) 3.30cpu seconds (kernel mode) 4.20s elapsed real time

MILESTONE 2



Runtimes: User 15.41, System 4.29, Elapsed 11.5

MILESTONE 3

```
# Running /usr/bin/time python m3.1.py
Loading fashion-mnist data... done
Loading model... done
New Inference
Op Time: 0.018376
Op Time: 0.070265
Correctness: 0.8397 Model: ece408
4.57user 3.17system 0:04.47elapsed 172%CPU (0avgtext+0avgdata 2828240maxresident)k
0i
nputs+4576outputs (0major+659360minor)pagefaults 0swaps
# Running python m3.1.py 100
Loading fashion-mnist data... done
Loading model... done
New Inference
Op Time: 0.000209
Op Time: 0.000696
Correctness: 0.84 Model: ece408
# Running python m3.1.py 1000
Loading fashion-mnist data... done
Loading model... done
New Inference
Op Time: 0.001868
Op Time: 0.006977
Correctness: 0.852 Model: ece408
# Running nvprof python m3.1.py
Loading fashion-mnist data... done
==546== NVPROF is profiling process 546, command: python m3.1.py
```

```
# Running nvprof python m3.1.py
Loading fashion-mnist data... done
==368== NVPROF is profiling process 368, command: python m3.1.py
Loading model... done
New Inference
Op Time: 0.018457
Op Time: 0.071863
Correctness: 0.8397 Model: ece408
==368== Profiling application: python m3.1.py
==368== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   78.58%  90.248ms         2  45.124ms  18.414ms  71.834ms  mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)
                   14.53%  16.693ms        20  834.63us  1.1200us  16.257ms  [CUDA memcpy HtoD]
                    2.18%  2.5048ms         2  1.2524ms  21.408us  2.4834ms  volta_sgemm_32x128_tn
                    2.11%  2.4244ms         2  1.2122ms  735.48us  1.6889ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tensor
<mshadow::gpu, int=4, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4, float>, float,
int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                    1.41%  1.6211ms         2  810.54us  21.888us  1.5992ms  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)
                    0.92%  1.0526ms         1  1.0526ms  1.0526ms  1.0526ms  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>, cud
nnTensorStruct*, cudnnPoolingStruct, float, cudnnPoolingStruct, int, cudnn::reduced_divisor, float)
                    0.14%  157.76us         1  157.76us  157.76us  157.76us  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)
                    0.07%  75.295us         1  75.295us  75.295us  75.295us  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)
                    0.02%  28.192us        13  2.1680us  1.1840us  6.5920us  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)
                    0.02%  24.032us         2  12.016us  2.5280us  21.504us  void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu,
int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::Broadcast1DExp<mshadow::Tensor<mshadow::gpu, int=1, float>, float, int=2, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shap
e<int=2>, int=2)
                    0.01%  11.455us        10  1.1450us     991ns  1.6960us  [CUDA memset]
                    0.01%  5.7600us         1  5.7600us  5.7600us  5.7600us  [CUDA memcpy DtoH]
                    0.00%  4.8630us         1  4.8630us  4.8630us  4.8630us  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::ReduceWithAxisExp<mshadow::red::maximum, mshadow::Tensor<mshadow::gpu, int=3, float>, float, int=3, bool=1, int=2>, float>>(mshadow
::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
      API calls:   42.94%  3.11387s        22  141.54ms  12.916us  1.62078s  cudaStreamCreateWithFlags
                   31.80%  2.30554s        22  104.80ms  90.633us  2.30082s  cudaMemGetInfo
                   21.39%  1.55128s        18  86.182ms     834ns  419.05ms  cudaFree
1.33%  96.657ms       912  105.98us     310ns  19.753ms  cudaFuncSetAttribute
                    1.28%  92.703ms         6  15.451ms  5.5350us  71.841ms  cudaDeviceSynchronize
                    0.47%  34.216ms         9  3.8018ms  24.097us  16.410ms  cudaMemcpy2DAsync
                    0.23%  16.579ms        66  251.19us  5.6160us  7.7308ms  cudaMalloc
                    0.19%  13.798ms       216  63.879us     891ns  12.162ms  cudaEventCreateWithFlags
```

MILESTONE 4

Optimization 1: Loop Unrolling with restrict variable and variable tuning

This is based on the concept that the fastest memory transfers and processing happen at the register level. By unrolling the loop into local registers, we surpass the speeds of using shared memory and increase thread activity. When memory accesses are independent, instead of doing 1 read at a time per thread, we can do many more reads per thread depending on the input size. This speeds things up especially when there is local data re-use. On the downside, memory in our registers is relatively much lower than any other storage in the GPU and CPU and that is our limitation.

Restricted is a pointer aliasing optimization. It is similar to const in a sense that the programmer makes an agreement with the program to never point to the same memory location from 2 different arrays. By doing so, we relieve the compiler from checking it's pointer accesses which boosts the performance speed of the kernel.

Optimization with all restricted variables and inner/middle for loop unrolls:

```
# Running /usr/bin/time python m3.1.py
Loading fashion-mnist data... done
Loading model... done
New Inference
B is: 10000 M is: 6 H is: 48 W is: 48 C is: 1 K is: 5 Op Time: 0.005552
B is: 10000 M is: 16 H is: 22 W is: 22 C is: 6 K is: 5 Op Time: 0.022083
Correctness: 0.8397 Model: ece408
4.37user 3.23system 0:04.27elapsed 177%CPU (0avgtext+0a
vgdata 2869136maxresident)k
0inputs+4568outputs (0major+669450minor)pagefaults 0swaps
# Running python m3.1.py 100
Loading fashion-mnist data... done
Loading model... done
New Inference
B is: 100 M is: 6 H is: 48 W is: 48 C is: 1 K is: 5 Op Time: 0.000081
B is: 100 M is: 16 H is: 22 W is: 22 C is: 6 K is: 5 Op Time: 0.000241
Correctness: 0.84 Model: ece408
# Running python m3.1.py 1000
Loading fashion-mnist data... done
Loading model... done
New Inference
 B is: 1000 M is: 6 H is: 48 W is: 48 C is: 1 K is: 5 Op Time: 0.000591
B is: 1000 M is: 16 H is: 22 W is: 22 C is: 6 K is: 5 Op Time: 0.002146
Correctness: 0.852 Model: ece408
# Running nvprof python m3.1.py
Loading model... done
New Inference
B is: 10000 M is: 6 H is: 48 W is: 48 C is: 1 K is: 5 Op Time: 0.005719
B is: 10000 M is: 16 H is: 22 W is: 22 C is: 6 K is: 5 Op Time: 0.022290
Correctness: 0.8397 Model: ece408
==548== Profiling application: python m3.1.py
==548== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   52.97%  27.921ms         2  13.960ms  5.6684ms  22.252ms  mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)
                   32.03%  16.884ms        20  844.21us  1.0880us  16.369ms  [CUDA memcpy HtoD]
                    4.75%  2.5025ms         2  1.2513ms  22.016us  2.4805ms  volta_sgemm_32x128_tn
                    4.60%  2.4245ms         2  1.2122ms  738.46us  1.6860ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tensor
<mshadow::gpu, int=4, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4, float>, float,
int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                    3.07%  1.6207ms         2  810.33us  22.048us  1.5986ms  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)
                    2.00%  1.0529ms         1  1.0529ms  1.0529ms  1.0529ms  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>, cud
nnTensorStruct*, cudnnPoolingStruct, float, cudnnPoolingStruct, int, cudnn::reduced_divisor, float)
                    0.30%  158.21us         1  158.21us  158.21us  158.21us  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)
                    0.14%  75.519us         1  75.519us  75.519us  75.519us  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)
                    0.05%  27.616us        13  2.1240us  1.1520us  6.5280us  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)
                    0.05%  23.872us         2  11.936us  2.5280us  21.344us  void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu,
int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::Broadcast1DExp<mshadow::Tensor<mshadow::gpu, int=1, float>, float, int=2, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shap
e<int=2>, int=2)
                    0.02%  11.200us        10  1.1200us     992ns  1.6000us  [CUDA memset]
                    0.01%  7.7440us         1  7.7440us  7.7440us  7.7440us  [CUDA memcpy DtoH]
                    0.01%  4.9280us         1  4.9280us  4.9280us  4.9280us  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::ReduceWithAxisExp<mshadow::red::maximum, mshadow::Tensor<mshadow::gpu, int=3, float>, float, int=3, bool=1, int=2>, float>>(mshadow
::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
      API calls:   43.39%  3.07026s        22  139.56ms  20.464us  1.58911s  cudaStreamCreateWithFlags
                   31.96%  2.26176s        22  102.81ms  89.756us  2.25666s  cudaMemGetInfo
                   21.85%  1.54582s        18  85.879ms     857ns  414.38ms  cudaFree
1.05%  74.158ms       912  81.313us     306ns  26.788ms  cudaFuncSetAttribute
                    0.48%  34.001ms         9  3.7779ms  16.677us  16.401ms  cudaMemcpy2DAsync
                    0.43%  30.368ms         6  5.0613ms  2.3660us  22.254ms  cudaDeviceSynchronize
                    0.20%  14.395ms        66  218.11us  5.7110us  5.2968ms  cudaMalloc
                    0.19%  13.655ms        12  1.1379ms  8.3940us  13.270ms  cudaMemcpy
```

Optimization with all restricted variables and inner for loop unroll:

```
# Running /usr/bin/time python m3.1.py
Loading fashion-mnist data... done
Loading model... done
New Inference
Op Time: 0.006589
Op Time: 0.016771
Correctness: 0.8397 Model: ece408
4.39user 3.37system 0:04.26elapsed 182%CPU (0avgtext+0avgdata 2845916maxresident)k
0inputs+4576outputs (0major+663406minor)pagefaults 0swaps
# Running nvprof python m3.1.py
Loading fashion-mnist data... done
==369== NVPROF is profiling process 369, command: python m3.1.py
Loading model... done
New Inference
Op Time: 0.006024
Op Time: 0.015141
Correctness: 0.8397 Model: ece408
==369== Profiling application: python m3.1.py
==369== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   46.24%  21.094ms         2  10.547ms  5.9786ms  15.115ms  mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)
                   37.26%  16.998ms        20  849.90us  1.0240us  16.461ms  [CUDA memcpy HtoD]
                    5.19%  2.3672ms         2  1.1836ms  720.95us  1.6462ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tensor
<mshadow::gpu, int=4, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4, float>, float,
int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                    4.91%  2.2386ms         2  1.1193ms  20.032us  2.2186ms  volta_sgemm_32x128_tn
                    3.54%  1.6151ms         2  807.55us  21.600us  1.5935ms  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)
                    2.23%  1.0191ms         1  1.0191ms  1.0191ms  1.0191ms  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>, cud
nnTensorStruct*, cudnnPoolingStruct, float, cudnnPoolingStruct, int, cudnn::reduced_divisor, float)
                    0.33%  149.95us         1  149.95us  149.95us  149.95us  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)
                    0.15%  67.967us         1  67.967us  67.967us  67.967us  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)
                    0.05%  24.991us        13  1.9220us  1.0560us  5.8880us  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)
                    0.05%  23.551us         2  11.775us  2.2720us  21.279us  void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu,
int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::Broadcast1DExp<mshadow::Tensor<mshadow::gpu, int=1, float>, float, int=2, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shap
e<int=2>, int=2)
                    0.02%  10.912us        10  1.0910us     928ns  1.8560us  [CUDA memset]
                    0.02%  7.6160us         1  7.6160us  7.6160us  7.6160us  [CUDA memcpy DtoH]
                    0.01%  4.5120us         1  4.5120us  4.5120us  4.5120us  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::ReduceWithAxisExp<mshadow::red::maximum, mshadow::Tensor<mshadow::gpu, int=3, float>, float, int=3, bool=1, int=2>, float>>(mshadow
```

Optimization with outer loop unroll:

```
# Running /usr/bin/time python m3.1.py
Loading fashion-mnist data... done
Loading model... done
New Inference
Op Time: 0.006358
Op Time: 0.015087
Correctness: 0.8397 Model: ece408
4.43user 3.32system 0:04.27elapsed 181%CPU (0avgtext+0avgdata 2830628maxresident)k
0inputs+4584outputs (0major
+659469minor)pagefaults 0swaps
# Running nvprof python m3.1.py
Loading fashion-mnist data... done
==369== NVPROF is profiling process 369, command: python m3.1.py
Loading model... done
New Inference
Op Time: 0.006474
Op Time: 0.015145
Correctness: 0.8397 Model: ece408
==369== Profiling application: python m3.1.py
==369== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   46.70%  21.545ms         2  10.772ms  6.4312ms  15.114ms  mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)
                   36.73%  16.949ms        20  847.45us  1.0560us  16.438ms  [CUDA memcpy HtoD]
                    5.15%  2.3740ms         2  1.1870ms  725.24us  1.6487ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tens
<mshadow::gpu, int=4, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4, float>, float
int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                    5.01%  2.3128ms         2  1.1564ms  21.280us  2.2916ms  volta_sgemm_32x128_tn
                    3.49%  1.6123ms         2  806.15us  22.016us  1.5903ms  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)
                    2.26%  1.0405ms         1  1.0405ms  1.0405ms  1.0405ms  void cudnn::detail::pooling_fw_4d_kernel<float, float, cudnn::detail::maxpooling_func<float, cudnnNanPropagati
_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>, c
nnTensorStruct*, cudnnPoolingStruct, float, cudnnPoolingStruct, int, cudnn::reduced_divisor, float)
                    0.33%  154.24us         1  154.24us  154.24us  154.24us  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tens
<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)
                    0.16%  75.327us         1  75.327us  75.327us  75.327us  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)
                    0.06%  27.968us        13  2.1510us  1.2160us  6.5280us  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)
                    0.05%  23.935us         2  11.967us  2.5600us  21.375us  void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu
int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::Broadcast1DExp<mshadow::Tensor<mshadow::gpu, int=1, float>, float, int=2, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Sh
e<int=2>, int=2)
                    0.02%  11.392us        10  1.1390us     960ns  1.9200us  [CUDA memset]
                    0.02%  7.8720us         1  7.8720us  7.8720us  7.8720us  [CUDA memcpy DtoH]
                    0.01%  5.0240us         1  5.0240us  5.0240us  5.0240us  void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu
```

Optimization with only x and k arrays restricted and inner for loop unroll:

```
Successfully installed mxnet
# Running /usr/bin/time python m3.1.py
Loading fashion-mnist data... done
Loading model... done
New Inference
Op Time: 0.006581
Op Time: 0.016771
Correctness: 0.8397 Model: ece408
4.47user 3.12system 0:04.29elapsed 176%CPU (0avgtext+0avgdata 2845384maxresident)k
0inputs+4576output
s (0major+663584minor)pagefaults 0swaps
# Running nvprof python m3.1.py
Loading fashion-mnist data... done
==370== NVPROF is profiling process 370, command: python m3.1.py
Loading model... done
New Inference
Op Time: 0.006699
Op Time: 0.016928
Correctness: 0.8397 Model: ece408
==370== Profiling application: python m3.1.py
==370== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   48.59%  23.552ms         2  11.776ms  6.6552ms  16.897ms  mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)
                   35.64%  17.276ms        20  863.81us  1.1200us  16.884ms  [CUDA memcpy HtoD]
                    4.88%  2.3665ms         2  1.1833ms  723.68us  1.6428ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tensor
<mshadow::gpu, int=4, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4, float>, float,
int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                    4.78%  2.3166ms         2  1.1583ms  20.672us  2.2959ms  volta_sgemm_32x128_tn
                    3.33%  1.6132ms         2  806.59us  22.239us  1.5909ms  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)
                    2.15%  1.0433ms         1  1.0433ms  1.0433ms  1.0433ms  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>, cud
nnTensorStruct*, cudnnPoolingStruct, float, cudnnPoolingStruct, int, cudnn::reduced_divisor, float)
                    0.32%  152.86us         1  152.86us  152.86us  152.86us  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)
                    0.16%  75.551us         1  75.551us  75.551us  75.551us  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)
                    0.06%  27.840us        13  2.1410us  1.1840us  6.4640us  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)
                    0.05%  23.872us         2  11.936us  2.5600us  21.312us  void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu,
int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::Broadcast1DExp<mshadow::Tensor<mshadow::gpu, int=1, float>, float, int=2, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shap
e<int=2>, int=2)
                    0.02%  11.264us        10  1.1260us     992ns  1.6000us  [CUDA memset]
                    0.01%  5.8560us         1  5.8560us  5.8560us  5.8560us  [CUDA memcpy DtoH]
```

Optimization with only y restriction:

```
# Running /usr/bin/time python m3.1.py
Loading fashion-mnist data... done
Loading model... done
New Inference
B is: 10000 M is: 6 H is: 48 W is: 48 C is: 1 K is: 5 Op Time: 0.006352
B is: 10000 M is: 16 H is: 22 W is: 22 C is: 6 K is: 5 Op Time: 0.015088
Correctness: 0.8397 Model: ece408
4.36user 3.41system 0:04.29elapsed 181%CPU (0avgtext+0avgdata 2828144maxresident)k
0inputs+4584outputs (0
major+659026minor)pagefaults 0swaps
# Running nvprof python m3.1.py
Loading fashion-mnist data... done
==368== NVPROF is profiling process 368, command: python m3.1.py
Loading model... done
New Inference
B is: 10000 M is: 6 H is: 48 W is: 48 C is: 1 K is: 5 Op Time: 0.006493
B is: 10000 M is: 16 H is: 22 W is: 22 C is: 6 K is: 5 Op Time: 0.015150
Correctness: 0.8397 Model: ece408
==368== Profiling application: python m3.1.py
==368== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   47.23%  21.553ms         2  10.777ms  6.4386ms  15.115ms  mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)
                   36.05%  16.453ms        20  822.63us  1.0560us  16.070ms  [CUDA memcpy HtoD]
                    5.19%  2.3693ms         2  1.1847ms  721.82us  1.6475ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<
<mshadow::gpu, int=4, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4,
int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                    5.05%  2.3043ms         2  1.1522ms  21.440us  2.2829ms  volta_sgemm_32x128_tn
                    3.53%  1.6122ms         2  806.12us  21.664us  1.5906ms  void op_generic_tensor_kernel<int=2, float, float, float, int=256, cudnnGenericOp_t=7, cudnnNanPr
cudnnDimOrder_t=0, int=1>(cudnnTensorStruct, float*, cudnnTensorStruct, float const *, cudnnTensorStruct, float const *, float, float, float, float, dimArray, reducedDivisorA
                    2.29%  1.0446ms         1  1.0446ms  1.0446ms  1.0446ms  void cudnn::detail::pooling_fw_4d_kernel<float, float, cudnn::detail::maxpooling_func<float, cudn
_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=
nnTensorStruct*, cudnnPoolingStruct, float, cudnnPoolingStruct, int, cudnn::reduced_divisor, float)
                    0.34%  153.41us         1  153.41us  153.41us  153.41us  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<
<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)
                    0.17%  75.359us         1  75.359us  75.359us  75.359us  void mshadow::cuda::SoftmaxKernel<int=8, float, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu,
 float>, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>>(mshadow::gpu, int=2, unsigned int)
                    0.06%  28.032us        13  2.1560us  1.1840us  6.4960us  void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor
int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::ScalarExp<float>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
                    0.05%  24.064us         2  12.032us  2.5600us  21.504us  void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tensor
int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::Broadcast1DExp<mshadow::Tensor<mshadow::gpu, int=1, float>, float, int=2, int=1>, float>>(mshadow::gpu, unsigned int
e<int=2>, int=2)
                    0.03%  11.776us        10  1.1770us     960ns  1.9840us  [CUDA memset]
                    0.01%  5.7600us         1  5.7600us  5.7600us  5.7600us  [CUDA memcpy DtoH]
                    0.01%  4.7040us         1  4.7040us  4.7040us  4.7040us  void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor
```

Size 8 tile width with all restricted array variables but no unrolling:

```
Successfully installed mxnet
✅ Running nvprof python m4.1.py
Loading fashion-mnist data... done
==279== NVPROF is profiling process 279, command: python m4.1.py
Loading model... done
New Inference
B is: 10000 M is: 6 H is: 48 W is: 48 C is: 1 K is: 5 Op Time: 0.011965
B is: 10000 M is: 16 H is: 22 W is: 22 C is: 6 K is: 5 Op Time: 0.028503
Correctness: 0.8397 Model: ece408
==279== Profiling application: python m4.1.py
==279== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   61.87%  40.373ms         2  20.187ms  11.910ms  28.463ms  mxnet::op::forward_kernel(float*, float con
st *, float const *, int, int, int, int, int, int)
                  26.00%  16.965ms        20  848.26us  1.0880us  16.417ms  [CUDA memcpy HtoD]
                   3.85%  2.5107ms         2  1.2554ms  20.928us  2.4898ms  volta_sgemm_32x128_tn
                   3.71%  2.4208ms         2  1.2104ms  734.27us  1.6865ms  void mshadow::cuda::MapPlanLargeKernel<msha
dow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=4, float>, float>, mshadow::expr
::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=
4, float>, float, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                   2.49%  1.6232ms         2  811.63us  21.888us  1.6014ms  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, reducedDivis
orArray)
                   1.62%  1.0546ms         1  1.0546ms  1.0546ms  1.0546ms  void cudnn::detail::pooling_fw_4d_kernel<fl
oat, float, cudnn::detail::maxpooling_func<float, cudnnNanPropagation_t=0>, int=0, bool=0>(cudnnTensorStruct, float cons
t *, cudnn::detail::pooling_fw_4d_kernel<float, float, cudnn::detail::maxpooling_func<float, cudnnNanPropagation_t=0>, i
nt=0, bool=0>, cudnnTensorStruct*, cudnnPoolingStruct, float, cudnnPoolingStruct, int, cudnn::reduced_divisor, float)
                   0.24%  157.76us         1  157.76us  157.76us  157.76us  void mshadow::cuda::MapPlanLargeKernel<msha
dow::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)
                   0.12%  75.072us         1  75.072us  75.072us  75.072us  void mshadow::cuda::SoftmaxKernel<int=8, fl
oat, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>, mshadow::expr::Plan<mshadow::Tensor<mshado
w::gpu, int=2, float>, float>>(mshadow::gpu, int=2, unsigned int)
```

Final Optimization: 2 loop unrolls, size 8 tile width, and all restricted array variables

```
✅ Running nvprof python m4.1.py
Loading fashion-mnist data... done
==278== NVPROF is profiling process 278, command: python m4.1.py
Loading model... done
New Inference
B is: 10000 M is: 6 H is: 48 W is: 48 C is: 1 K is: 5 Op Time: 0.009558
B is: 10000 M is: 16 H is: 22 W is: 22 C is: 6 K is: 5 Op Time: 0.020876
Correctness: 0.8397 Model: ece408
==278== Profiling application: python m4.1.py
==278== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   55.33%  30.366ms         2  15.183ms  9.5152ms  20.851ms  mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)
                  30.74%  16.868ms        20  843.41us  1.0880us  16.462ms  [CUDA memcpy HtoD]
                   4.33%  2.3775ms         2  1.1888ms  728.09us  1.6494ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan
<mshadow::gpu, int=4, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4,
int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                   4.20%  2.3065ms         2  1.1533ms  21.344us  2.2852ms  volta_sgemm_32x128_tn
                   2.94%  1.6127ms         2  806.36us  21.280us  1.5914ms  void op_generic_tensor_kernel<int=2, float, float, float, int=256, cudnnGenericOp_t=7, cudnnNanP
cudnnDimOrder_t=0, int=1>(cudnnTensorStruct, float*, cudnnTensorStruct, float const *, cudnnTensorStruct, float const *, float, float, float, float, dimArray, reducedDivisor
                   1.91%  1.0457ms         1  1.0457ms  1.0457ms  1.0457ms  void cudnn::detail::pooling_fw_4d_kernel<float, float, cudnn::detail::maxpooling_func<float, cud
_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
nnTensorStruct*, cudnnPoolingStruct, float, cudnnPoolingStruct, int, cudnn::reduced_divisor, float)
                   0.28%  151.68us         1  151.68us  151.68us  151.68us  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan
<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)
                   0.14%  75.199us         1  75.199us  75.199us  75.199us  void mshadow::cuda::SoftmaxKernel<int=8, float, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu
 float>, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>>(mshadow::gpu, int=2, unsigned int)
                   0.05%  28.032us        13  2.1560us  1.2160us  6.4320us  void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tenso
int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::ScalarExp<float>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
                   0.04%  23.872us         2  11.936us  2.5600us  21.312us  void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tenso
int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::Broadcast1DExp<mshadow::Tensor<mshadow::gpu, int=1, float>, float, int=2, int=1>, float>>(mshadow::gpu, unsigned in
e<int=2>, int=2)
                   0.02%  11.424us        10  1.1420us     992ns  1.9840us  [CUDA memset]
                   0.01%  5.7280us         1  5.7280us  5.7280us  5.7280us  [CUDA memcpy DtoH]
                   0.01%  4.8320us         1  4.8320us  4.8320us  4.8320us  void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tenso
int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::ReduceWithAxisExp<mshadow::red::maximum, mshadow::Tensor<mshadow::gpu, int=3, float>, float, int=3, bool=1, int=2>,
::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
      API calls:   42.72%  2.99088s        22  135.95ms  14.317us  1.53094s  cudaStreamCreateWithFlags
                   33.84%  2.36879s        22  107.67ms  90.835us  2.36399s  cudaMemGetInfo
                   21.10%  1.47682s        18  82.046ms     826ns  392.91ms  cudaFree
0.88%  61.723ms       912  67.678us     301ns  18.747ms  cudaFuncSetAttribute
```

Analysis:

Original op-time: .018, .072

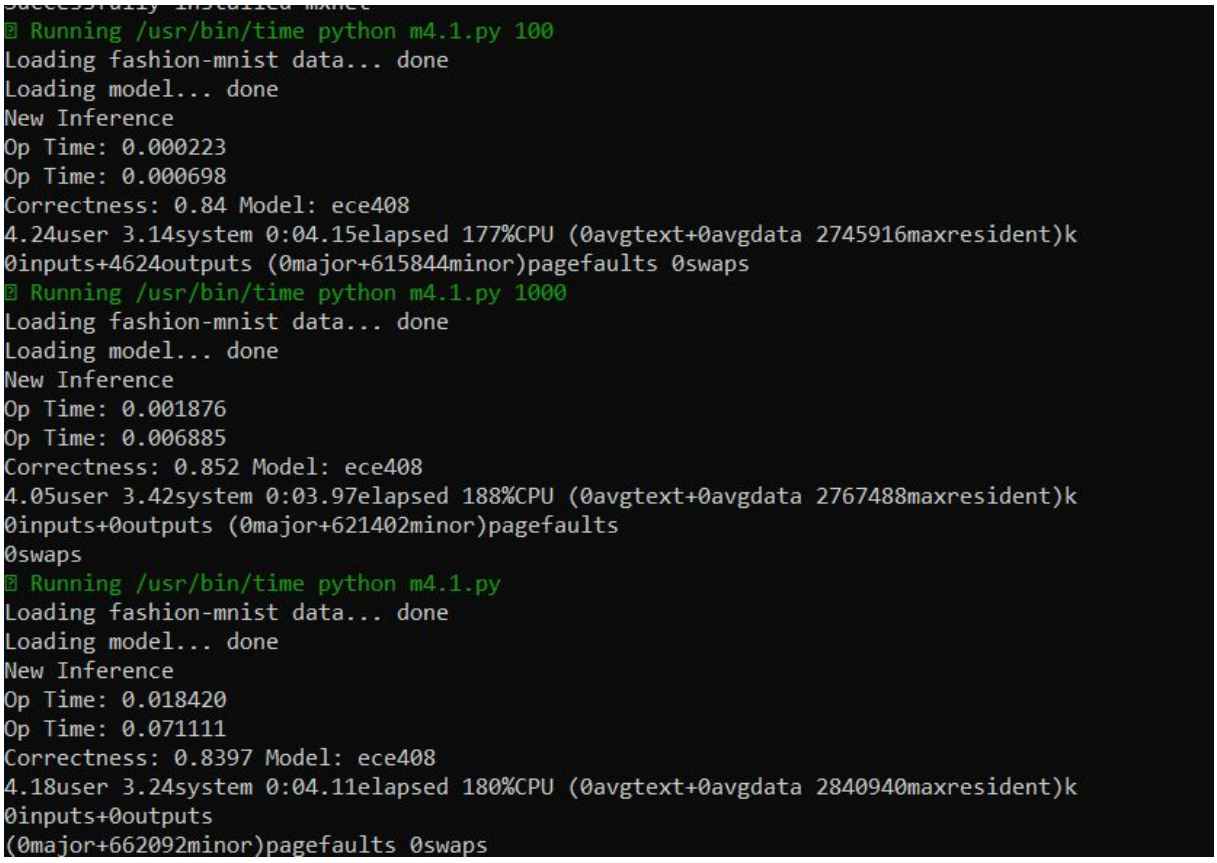
Post optimization op-time: .009, .021

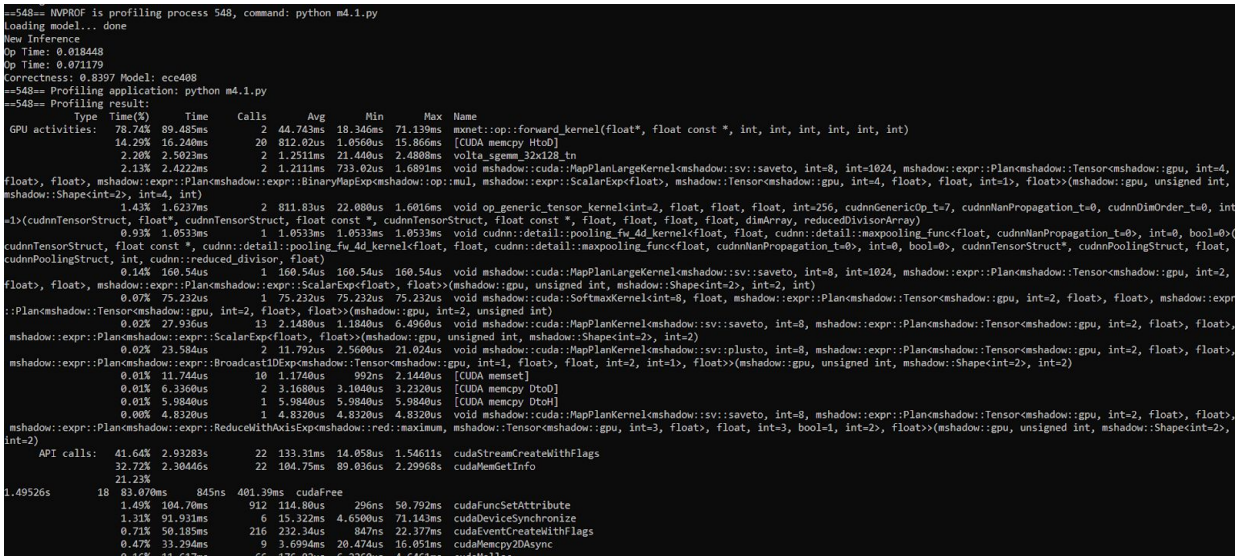
Unrolling seems to allow for more gpu compute time and decrease the op time. This is because we reuse the variable y[] so many times and unrolling allows us to be faster than the

usual iterative indexing process. Using restrict on the variables also improved performance. Furthermore, changing the tile width to 8 really boosted the block utilizations.

Optimization 2: Weight Matrix (kernel values) in Constant Memory

The outputs of the neural network are calculated by convoluting the input matrices with the a weight matrix, which remains constant throughout the entirety of the execution. In milestone 3, the weight matrix pointer is passed into the forward_kernel along with the input and output matrix pointers. Whenever the forward_kernel needs values from the weight matrix, it must access it through global memory, which is one of the slowest device memories in CUDA. This optimization places the weight matrix into constant memory instead, speeding up the access times for weight kernels.





Analysis (For the default dataset containing 10000 images):
Milestone 3 Optimes: 0.1847, 0.07168
Post Optimization Optimes: 0.1842, 0.071179

Optimization 3: Shared Memory Convolution

For forward convolution, we noticed that each block corresponds with a tile within one input image convolved with one mask, which means that for every thread in one block, it needs to perform the same global memory reads for k*k elements in that mask, and it would be much faster if we divide the loading mask into shared memory to each thread and make the process parallel, which would give a shorter time for memory reads for each block since shared memory reads are faster than global memory reads.

For each block, since it performs the convolution for a tile of one input image, we use the tiling optimization methods for preloading the tile of input elements into the shared memory with all threads in parallel and make the access to the input element faster by reading from shared memory rather than from global memory.

Thus this optimization reduces the memory read times and speed up the kernel by utilizing shared memory properties.

-- analysis:
Milestone 3 without optimization : Op Time: 0.018446 - Op Time: 0.071832
After optimization : Op Time: 0.006718 - Op Time: 0.034986

```
  Running setup.py develop for mxnet
Successfully installed mxnet
* Running nvprof python m4.1.py
Loading fashion-mnist data... done
==279== NVPROF is profiling process 279, command: python m4.1.py
Loading model... done
New Inference
Op Time: 0.006718
Op Time: 0.034986
Correctness: 0.8397 Model: ece408
==279== Profiling application: python m4.1.py
==279== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   63.04%  41.632ms         2  20.816ms  6.6698ms  34.962ms  mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)
                   28.00%  18.500ms        20  925.43us  1.1200us  16.115ms  [CUDA memcpy HtoD]
                    3.78%  2.4953ms         2  1.2476ms  21.216us  2.4741ms  volta_sgemm_32x128_tn
                    3.66%  2.4191ms         2  1.2096ms  736.67us  1.6825ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=4, float>, float>, mshadow::ex
pr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4, float>, float, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                    2.46%  1.6224ms         2  811.18us  22.176us  1.6002ms  void op_generic_tensor_kernel<int=2, float, float, float, int=256, cudnnGenericOp_t=7, cudnnNanPropagation_t=0, cudnnDimOrder_t=0, int=1>(cudnnTensorStruct, floa
t*, cudnnTensorStruct, float const *, cudnnTensorStruct, float const *, float, float, float, dimArray, reducedDivisorArray)
                    1.60%  1.0552ms         1  1.0552ms  1.0552ms  1.0552ms  void cudnn::detail::pooling_fw_4d_kernel<float, float, cudnn::detail::maxpooling_func<float, cudnnNanPropagation_t=0>, int=0, bool=0>(cudnnTensorStruct, float co
nst *, 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
)
                    0.24%  158.40us         1  158.40us  158.40us  158.40us  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>, mshadow::ex
pr::Plan<mshadow::expr::ScalarExp<float>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2, int)
                    0.11%  75.135us         1  75.135us  75.135us  75.135us  void mshadow::cuda::SoftmaxKernel<int=8, float, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>, mshadow::expr::Plan<mshadow::Tensor<msha
dow::gpu, int=2, float>, float>>(mshadow::gpu, int=2, unsigned int)
                    0.04%  27.840us        13  2.1410us  1.1840us  6.4960us  void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>, mshadow::expr::Plan<mshado
w::expr::ScalarExp<float>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
                    0.04%  23.903us         2  11.951us  2.5600us  21.343us  void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>, mshadow::expr::Plan<mshado
w::expr::BroadcastIDExp<mshadow::Tensor<mshadow::gpu, int=1, float>, float, int=2, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
                    0.02%  11.135us        10  1.1130us     991ns  1.6320us  [CUDA memset]
                    0.01%  5.7920us         1  5.7920us  5.7920us  5.7920us  [CUDA memcpy DtoH]
                    0.01%  4.7030us         1  4.7030us  4.7030us  4.7030us  void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>, mshadow::expr::Plan<mshado
w::expr::ReduceWithAxisExp<mshadow::red::maximum, mshadow::Tensor<mshadow::gpu, int=3, float>, float, int=3, bool=1, int=2>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
      API calls:   41.34%  2.9637ms        22  134.71us  14.319us  1.5549ms  cudaStreamCreateWithFlags
                   33.94%  2.4332ms        22  110.60ms  89.696us  2.42654s  cudaMemGetInfo
                   21.10%  1.51264s        18  84.036ms     859ns  402.51ms  cudaFree
1.35%  96.713ms         912  106.05us     312ns  53.475ms  cudaFuncSetAttribute
                    0.61%  44.073ms         6  7.3454ms  2.3540us  34.965ms  cudaDeviceSynchronize
                    0.58%  41.661ms       216  192.88us     899ns  38.522ms  cudaEventCreateWithFlags
                    0.47%  33.857ms         9  3.7619ms  20.480us  16.302ms  cudaMemcpy2DAsync
                    0.22%  15.535ms        66  235.38us  6.8090us  3.8565ms  cudaMalloc
                    0.11%  8.2321ms         4  2.0580ms  422.81us  3.4506ms  cudaGetDeviceProperties
                    0.07%  4.7598ms        29  164.13us  2.0560us  2.2415ms  cudaStreamSynchronize
                    0.05%  3.5854ms        12  298.78us  7.3750us  3.1627ms  cudaMemcpy
                    0.05%  3.4543ms         4  864.08us  60.236us  1.6834ms  cuDeviceGetName
                    0.03%  2.4730ms       375  6.5940us     286ns  329.11us  cuDeviceGetAttribute
                    0.01%  937.47us         8  117.18us  13.654us  722.28us  cudaStreamCreateWithPriority
                    0.01%  732.13us         2  366.07us  48.250us  683.88us  cudaHostAlloc
                    0.01%  616.85us         4  154.21us  92.762us  274.91us  cuDeviceTotalMem
                    0.01%  590.00us         4  147.50us  77.685us  253.49us  cudaStreamCreate
                    0.01%  544.84us        27  20.179us  7.9750us  65.391us  cudaLaunchKernel
                    0.00%  318.63us        10  31.863us  8.0650us  111.01us  cudaMemsetAsync
                    0.00%  292.23us       202  1.4460us     566ns  4.1240us  cudaDeviceGetAttribute
                    0.00%  150.71us        29  5.1960us  1.0150us  16.833us  cudaSetDevice
                    0.00%  106.03us       557     190ns      78ns     830ns  cudaGetLastError
                    0.00%  68.724us         4  17.181us  1.6760us  57.460us  cudaEventRecord
                    0.00%  46.950us        18  2.6080us     678ns  8.6310us  cudaGetDevice
                    0.00%  36.583us         6  6.0970us  1.1820us  13.574us  cudaEventCreate
                    0.00%  7.4950us         2  3.7470us  2.6290us  4.8660us  cudaHostGetDevicePointer
                    0.00%  6.3520us         2  3.1760us  3.1540us  3.1980us  cudaEventQuery
                    0.00%  5.5900us         2  2.7950us  1.5700us  3.9360us  cudaDeviceGetStreamPriorityRange
                    0.00%  5.3470us        20     267ns     103ns     573ns  cudaPeekAtLastError
                    0.00%  4.6950us         6     782ns     387ns  1.6070us  cuDeviceGetCount
                    0.00%  4.1670us         3  1.3890us     796ns  2.4950us  cuInit
                    0.00%  3.9850us         1  3.9850us  3.9850us  3.9850us  cuDeviceGetPCIBusId
                    0.00%  3.7680us         5     753ns     413ns  1.2630us  cuDeviceGet
                    0.00%  3.1300us         4     782ns     526ns  1.2800us  cudaGetDeviceCount
                    0.00%  2.6230us         4     655ns     351ns  1.1990us  cuDeviceGetUuid
                    0.00%  1.9350us         3     645ns     303ns  1.2670us  cuDriverGetVersion
* The build folder has been uploaded to http://s3.amazonaws.com/files.rai-project.com/userdata/build-5cb8cba97c204654eea2fb19.tar.gz. The data will be present for only a short duration of time.
* Server has ended your request.
wirelessprv-10-192-203-94:uiuc xinyiguo$
```

```
* Running pip2 install --user -e /mxnet/python
DEPRECATION: Python 2.7 will reach the end of its life on January 1st, 2020. Please upgrade your Python as Python 2.7 won't be maintained after that date. A future version of pip will drop support
for Python 2.7.
Obtaining file:///mxnet/python
Requirement already satisfied: graphviz<0.9.0,>=0.8.1 in /root/.local/lib/python2.7/site-packages (from mxnet==1.3.1) (0.8.4)
Requirement already satisfied: numpy<=1.15.0,>=1.8.2 in /root/.local/lib/python2.7/site-packages (from mxnet==1.3.1) (1.14.0)
Requirement already satisfied: requests<2.19.0,>=2.18.4 in /root/.local/lib/python2.7/site-packages (from mxnet==1.3.1) (2.18.4)
Requirement already satisfied: idna<2.7,>=2.5 in /root/.local/lib/python2.7/site-packages (from requests<2.19.0,>=2.18.4->mxnet==1.3.1) (2.6)
Requirement already satisfied: urllib3<1.23,>=1.21.1 in /root/.local/lib/python2.7/site-packages (from requests<2.19.0,>=2.18.4->mxnet==1.3.1) (1.22)
Requirement already satisfied: certifi>=2017.4.17 in /root/.local/lib/python2.7/site-packages (from requests<2.19.0,>=2.18.4->mxnet==1.3.1) (2018.11.29)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /root/.local/lib/python2.7/site-packages (from requests<2.19.0,>=2.18.4->mxnet==1.3.1) (3.0.4)
Installing collected packages: mxnet
  Running setup.py develop for mxnet
Successfully installed mxnet
* Running nvprof -o timeline.nvprof python m3.1.py
Loading fashion-mnist data... done
==280== NVPROF is profiling process 280, command: python m3.1.py
Loading model... done
New Inference
Op Time: 0.006707
Op Time: 0.034936
Correctness: 0.8397 Model: ece408
==280== Generated result file: /build/timeline.nvprof
* Running nvprof --kernels "::forward:1" --analysis-metrics -o forward1_analysis.nvprof python m3.1.py
Loading fashion-mnist data... done
==377== NVPROF is profiling process 377, command: python m3.1.py
Loading model... done
New Inference
==377== Some kernel(s) will be replayed on device 0 in order to collect all events/metrics.
Replaying kernel "mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)" (done)
Op Time: 7.356935l events
Op Time: 0.031711
Correctness: 0.8397 Model: ece408
==377== Generated result file: /build/forward1_analysis.nvprof
* Running nvprof --kernels "::forward:2" --analysis-metrics -o forward2_analysis.nvprof python m3.1.py
Loading fashion-mnist data... done
==475== NVPROF is profiling process 475, command: python m3.1.py
Loading model... done
New Inference
Op Time: 0.006676
==475== Some kernel(s) will be replayed on device 0 in order to collect all events/metrics.
Replaying kernel "mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)" (done)
Op Time: 36.194178 events
Correctness: 0.8397 Model: ece408
==475== Generated result file: /build/forward2_analysis.nvprof
* The build folder has been uploaded to http://s3.amazonaws.com/files.rai-project.com/userdata/build-5cb8edb27c2046560458c4c1.tar.gz. The data will be present for only a short duration of time.
* Server has ended your request.
wirelessprv-10-192-203-94:uiuc xinyiguo$
```

NVVP Analysis:

Milestone 3 (Unoptimized)

Occupancy Is Not Limiting Kernel Performance

The kernel's block size, register usage, and shared memory usage allow it to fully utilize all warps on the GPU. More...

Variable	Achieved	Theoretical	Device Limit	Grid Size: [10000,6,9] (540000 blocks)Block Size: [16,16,1] (256 threads)
Occupancy Per SM				
Active Blocks		8	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Active Warps	63.46	64	64	0 5 10 15 20 25 30 35 40 45 50 55 60 64
Active Threads		2048	2048	0 256 512 768 1024 1280 1536 1792 2048
Occupancy	99.2%	100%	100%	0% 15% 30% 45% 60% 75% 90% 100%
Warps				
Threads/Block		256	1024	0 128 256 384 512 640 768 896 1024
Warps/Block		8	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Block Limit		8	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Registers				
Registers/Thread		28	65536	0 8192 16384 24576 32768 40960 49152 57344 65536
Registers/Block		8192	65536	0 16k 32k 48k 64k
Block Limit		8	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Shared Memory				
Shared Memory/Block		0	98304	0 16k 32k 48k 64k 80k 96k
Block Limit		0	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32

Memory Bandwidth And Utilization

The following table shows the memory bandwidth used by this kernel for the various types of memory on the device. The table also shows the utilization of each memory type relative to the maximum throughput supported by the memory. More...

	Transactions	Bandwidth	Utilization
Shared Memory			
Shared Loads	0	0 B/s	
Shared Stores	0	0 B/s	
Shared Total	0	0 B/s	Idle Low Medium High Max
L2 Cache			
Reads	27256512	52.557 GB/s	
Writes	480480498	926.477 GB/s	
Total	507737010	979.034 GB/s	Idle Low Medium High Max
Unified Cache			
Local Loads	0	0 B/s	
Local Stores	0	0 B/s	
Global Loads	499476236	963.105 GB/s	
Global Stores	480480000	926.476 GB/s	
Texture Reads	128550696	991.501 GB/s	
Unified Total	1108506932	2,881.082 GB/s	Idle Low Medium High Max
Device Memory			
Reads	33603390	64.795 GB/s	
Writes	21577031	41.605 GB/s	
Total	55180421	106.401 GB/s	Idle Low Medium High Max
System Memory [PCIe configuration: Gen3 x8, 8 Gbit/s]			
Reads	0	0 B/s	Idle Low Medium High Max
Writes	5	9.641 kB/s	Idle Low Medium High Max

Constant Memory

Results

Occupancy Is Not Limiting Kernel Performance

The kernel's block size, register usage, and shared memory usage allow it to fully utilize all warps on the GPU. More...

Variable	Achieved	Theoretical	Device Limit	Grid Size: [10000,6,9] (540000 blocks)Block Size: [16,16,1] (256 threa
Occupancy Per SM				
Active Blocks		8	32	0 3 6 9 12 15 18 21 24 27 30 32
Active Warps	63.46	64	64	0 6 12 18 24 30 36 42 48 54 60 64
Active Threads		2048	2048	0 256 512 768 1024 1280 1536 1792 2048
Occupancy	99.2%	100%	100%	0% 25% 50% 75% 100%
Warps				
Threads/Block		256	1024	0 128 256 384 512 640 768 896 1024
Warps/Block		8	32	0 3 6 9 12 15 18 21 24 27 30 32
Block Limit		8	32	0 3 6 9 12 15 18 21 24 27 30 32
Registers				
Registers/Thread		28	65536	0 8192 16384 24576 32768 40960 49152 57344 65536
Registers/Block		8192	65536	0 16k 32k 48k 64k
Block Limit		8	32	0 3 6 9 12 15 18 21 24 27 30 32
Shared Memory				
Shared Memory/Block		0	98304	0 16k 32k 48k 64k 80k 96k
Block Limit		0	32	0 3 6 9 12 15 18 21 24 27 30 32

Memory Bandwidth And Utilization

The following table shows the memory bandwidth used by this kernel for the various types of memory on the device. The table also shows the utilization of each memory type relative to the maximum throughput supported by the memory. More...

	Transactions	Bandwidth	Utilization
Shared Memory			
Shared Loads	0	0 B/s	
Shared Stores	0	0 B/s	
Shared Total	0	0 B/s	Idle Low Medium High Max
L2 Cache			
Reads	27276447	51.967 GB/s	
Writes	480480102	915.404 GB/s	
Total	507756549	967.371 GB/s	Idle Low Medium High Max
Unified Cache			
Local Loads	0	0 B/s	
Local Stores	0	0 B/s	
Global Loads	499475103	951.593 GB/s	
Global Stores	480480000	915.404 GB/s	
Texture Reads	128563020	979.746 GB/s	
Unified Total	1108518123	2,846.744 GB/s	Idle Low Medium High Max
Device Memory			
Reads	33602900	64.02 GB/s	
Writes	21601630	41.155 GB/s	
Total	55204530	105.175 GB/s	Idle Low Medium High Max
System Memory [PCIe configuration: Gen3 x8, 8 Gbit/s]			
Reads	0	0 B/s	Idle Low Medium High Max
Writes	5	9.525 kB/s	Idle Low Medium High Max

Unrolling

Occupancy Is Not Limiting Kernel Performance

The kernel's block size, register usage, and shared memory usage allow it to fully utilize all warps on the GPU. More...

Variable	Achieved	Theoretical	Device Limit	Grid Size: [10000,6,36] (2160000 blocks)Block Size: [8,8,1] (64 threads)
Occupancy Per SM				
Active Blocks		32	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Active Warps	42.15	64	64	0 5 10 15 20 25 30 35 40 45 50 55 60 64
Active Threads		2048	2048	0 256 512 768 1024 1280 1536 1792 2048
Occupancy	65.9%	100%	100%	0% 15% 30% 45% 60% 75% 90% 100%
Warps				
Threads/Block		64	1024	0 128 256 384 512 640 768 896 1024
Warps/Block		2	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Block Limit		32	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Registers				
Registers/Thread		32	65536	0 8192 16384 24576 32768 40960 49152 57344 65536
Registers/Block		2048	65536	0 16k 32k 48k 64k
Block Limit		32	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Shared Memory				
Shared Memory/Block		0	98304	0 16k 32k 48k 64k 80k 96k
Block Limit		0	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32

Memory Bandwidth And Utilization

The following table shows the memory bandwidth used by this kernel for the various types of memory on the device. The table also shows the utilization of each memory type relative to the maximum throughput supported by the memory. More...

	Transactions	Bandwidth	Utilization
Shared Memory			
Shared Loads	0	0 B/s	
Shared Stores	0	0 B/s	
Shared Total	0	0 B/s	Idle Low Medium High Max
L2 Cache			
Reads	46994088	136.784 GB/s	
Writes	22440016	65.316 GB/s	
Total	69434104	202.1 GB/s	Idle Low Medium High Max
Unified Cache			
Local Loads	0	0 B/s	
Local Stores	0	0 B/s	
Global Loads	757622591	2,205.192 GB/s	
Global Stores	22440000	65.316 GB/s	
Texture Reads	341576365	3,976.869 GB/s	
Unified Total	1121638956	6,247.376 GB/s	Idle Low Medium High Max
Device Memory			
Reads	66173944	192.611 GB/s	
Writes	22486769	65.452 GB/s	
Total	88660713	258.062 GB/s	Idle Low Medium High Max
System Memory [PCIe configuration: Gen3 x16, 8 Gbit/s]			
Reads	0	0 B/s	Idle Low Medium High Max
Writes	5	14.553 kB/s	Idle Low Medium High Max

Shared Convolution

Memory Bandwidth And Utilization

The following table shows the memory bandwidth used by this kernel for the various types of memory on the device. The table also shows the utilization of each memory type relative to the maximum throughput supported by the memory. More..

	Transactions	Bandwidth	Utilization
Shared Memory			
Shared Loads	330191009	6,995.915 GB/s	
Shared Stores	15161650	321.237 GB/s	
Shared Total	345352659	7,317.152 GB/s	Idle Low Medium High Max
L2 Cache			
Reads	26882967	142.396 GB/s	
Writes	18480016	97.886 GB/s	
Total	45362983	240.282 GB/s	Idle Low Medium High Max
Unified Cache			
Local Loads	0	0 B/s	
Local Stores	0	0 B/s	
Global Loads	26891524	142.441 GB/s	
Global Stores	18480000	97.886 GB/s	
Texture Reads	235145723	4,982.145 GB/s	
Unified Total	280517247	5,222.472 GB/s	Idle Low Medium High Max
Device Memory			
Reads	33594060	177.943 GB/s	
Writes	18517670	98.086 GB/s	
Total	52111730	276.029 GB/s	Idle Low Medium High Max
System Memory [PCIe configuration: Gen3 x8, 8 Gbit/s]			
Reads	0	0 B/s	Idle Low Medium High Max
Writes	5	26.484 kB/s	Idle Low Medium High Max

Occupancy Per SM				
Active Blocks		6	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Active Warps	46.65	48	64	0 5 10 15 20 25 30 35 40 45 50 55 60 64
Active Threads		1536	2048	0 256 512 768 1024 1280 1536 1792 2048
Occupancy	72.9%	75%	100%	0% 15% 30% 45% 60% 75% 90% 100%
Warps				
Threads/Block		256	1024	0 128 256 384 512 640 768 896 1024
Warps/Block		8	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Block Limit		8	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Registers				
Registers/Thread		39	65536	0 8192 16384 24576 32768 40960 49152 57344 65536
Registers/Block		10240	65536	0 16k 32k 48k 64k
Block Limit		6	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Shared Memory				
Shared Memory/Block		1700	98304	0 16k 32k 48k 64k 80k 96k
Block Limit		54	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32

Based on the NVVP analysis shown above, we can see that constant memory has similar statistics as the milestone 3 kernel, and offers very little performance boosts. Meanwhile,

unrolling provides a lot more, as can be seen by the amount of active threads and warps, which is maxed out. Finally, shared memory convolution is also an effective enhancement -- since shared memory one of the fastest memories, using it for computation greatly enhances performance.

MILESTONE 5:

Optimization 4: Variable Sweeping

Sometimes, the best way to find the best optimization is by trying every combination of tuning based off of many possible areas of improvement or realizing the effect of certain variables on the optime.We started off by doing a variable sweep and tile difference for shared convolutions because some of the optimes were the lowest we'd had. When we tried introducing new kernels sometimes, we might get a slower run time. However, if we adjusted the tile widths and switched out some unroll numbers, we could sometimes halve the time it took. Additionally, we restricted the array pointers because read-only memory is faster than normal global memory.

	Tile width 4	Tile width 8	Tile width 16	Tilewidth 24	Tilewidth 32
Shared memory conv	Op Time: 0.021967 Op Time: 0.042524	Op Time: 0.008990 Op Time: 0.020861	Op Time: 0.005638 Op Time: 0.028150	Op Time: 0.007272 Op Time: 0.020014	Op Time: 0.015271 Op Time: 0.036846
Kernel Fusion	Op Time: 0.030086 Op Time: 0.050335	Op Time: 0.007826 Op Time: 0.015902	Op Time: 0.010030 Op Time: 0.015930	Op Time: 0.015342 Op Time: 0.014376	Op Time: 0.024249 Op Time: 0.019810

nvprof Analysis: Sweeping helps us find the optimal tile widths for resource utilization

Optimization 5: Unroll Matrix Multiply Fusion (w/Constant Kernel)

We expected Optimization 4 to help much more than it actually did which lead us to fusing the kernels. Due to the fact that calling multiple kernels for one pass involves many transfers of memory, the original unroll/Matrix multiply method will not be able to perform at it's optimized time. By fusing the two kernels together, we eliminate the transfers of memory to 2 separate kernels. Additionally, by adding the filter to Constant memory, we hope to optimize the memory transfers when we load the filter values.

```
Loading fashion-mnist data... done
==373== NVPROF is profiling process 373, command: python m4.1.py
Loading model... done
New Inference
Op Time: 0.017461
Op Time: 0.011492
Correctness: 0.8397 Model: ece408
==373== Profiling application: python m4.1.py
==373== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   53.56%  28.879ms         2  14.440ms  11.465ms  17.415ms  mxnet::op::forward_kernel2(float*, float const *, float const *, int, int, int, int, int, int)
                   31.73%  17.105ms        20  855.27us  1.0560us  16.571ms  [CUDA memcpy HtoD]
                    4.65%  2.5057ms         2  1.2528ms  22.015us  2.4837ms  volta_sgemm_32x128_tn
                    4.51%  2.4320ms         2  1.2160ms  735.61us  1.6964ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=4, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4, float>, float, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                    3.03%  1.6347ms         2  817.37us  22.048us  1.6127ms  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)
                    1.96%  1.0546ms         1  1.0546ms  1.0546ms  1.0546ms  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)
                    0.28%  153.38us         1  153.38us  153.38us  153.38us  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)
                    0.14%  75.360us         1  75.360us  75.360us  75.360us  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)
                    0.05%  27.872us        13  2.1440us  1.1520us  6.5280us  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)
                    0.04%  24.096us         2  12.048us  2.4960us  21.600us  void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::Broadcast1DExp<mshadow::Tensor<mshadow::gpu, int=1, float>, float, int=2, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
                    0.02%  12.000us        10  1.2000us     992ns  1.9840us  [CUDA memset]
                    0.01%  7.7760us         1  7.7760us  7.7760us  7.7760us  [CUDA memcpy DtoH]
                    0.01%  5.1840us         1  5.1840us  5.1840us  5.1840us  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::ReduceWithAxisExp<mshadow::red::maximum, mshadow::Tensor<mshadow::gpu, int=3, float>, float, int=3, bool=1, int=2>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
      API calls:   42.12%  3.13130ms        22  142.33ms  14.260us  1.61166s  cudaStreamCreateWithFlags
                   32.06%  2.38323ms        22  108.33ms  90.308us  2.37832s  cudaMemGetInfo
                   23.30%  1.73172s         18  96.207ms     824ns  448.14ms  cudaFree
1.01%  74.746ms       912  81.958us     309ns  45.990ms  cudaFuncSetAttribute
                    0.46%  34.396ms         9  3.8218ms  28.493us  16.606ms  cudaMemcpy2DAsync
                    0.42%  31.339ms         6  5.2231ms  4.5980us  17.419ms  cudaDeviceSynchronize
                    0.18%  13.572ms       216  62.831us     875ns  12.383ms  cudaEventCreateWithFlags
                    0.13%  9.3435ms        66  141.57us  6.0200us  1.9787ms  cudaMalloc
```

NVVP:

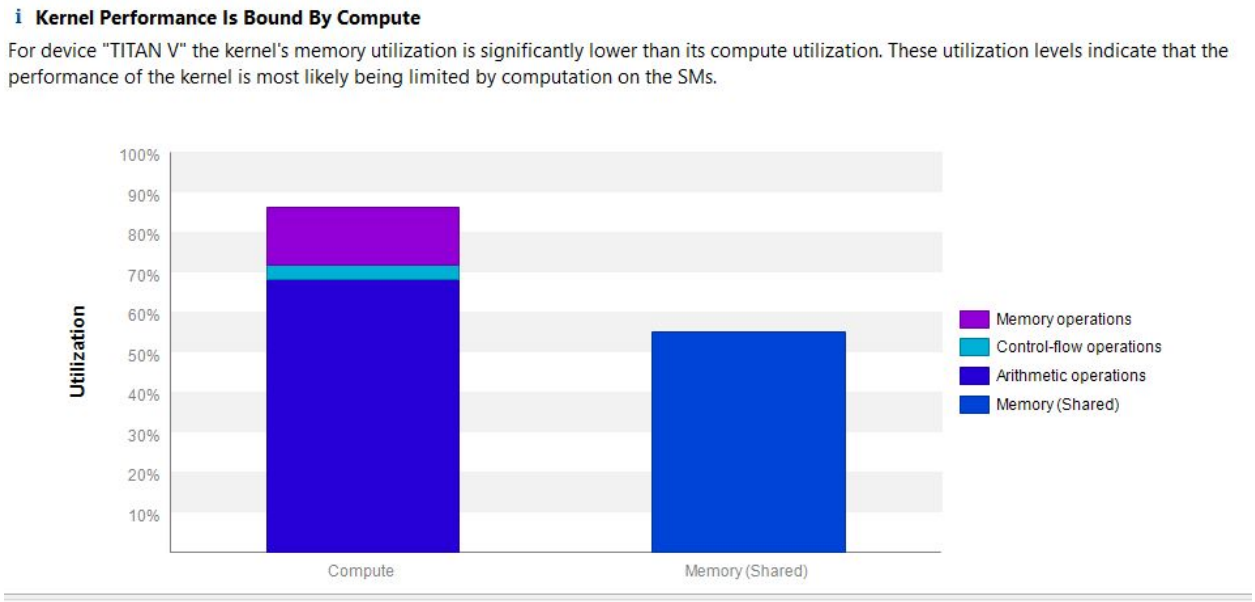
Results

	Transactions	Bandwidth	Utilization
Shared Memory			
Shared Loads	331854593	8,894.192 GB/s	
Shared Stores	15175163	406.717 GB/s	
Shared Total	347029756	9,300.908 GB/s	Idle Low Medium High Max
L2 Cache			
Reads	26882050	180.12 GB/s	
Writes	18480247	123.824 GB/s	
Total	45362297	303.944 GB/s	Idle Low Medium High Max
Unified Cache			
Local Loads	0	0 B/s	
Local Stores	0	0 B/s	
Global Loads	26909714	180.305 GB/s	
Global Stores	18480000	123.823 GB/s	
Texture Reads	235145991	6,302.259 GB/s	
Unified Total	280535705	6,606.387 GB/s	Idle Low Medium High Max
Device Memory			
Reads	33594036	225.092 GB/s	
Writes	18516349	124.066 GB/s	
Total	52110385	349.159 GB/s	Idle Low Medium High Max
System Memory [PCIe configuration: Gen3 x8, 8 Gbit/s]			
Reads	0	0 B/s	Idle Low Medium High Max
Writes	5	33.501 kB/s	Idle Low Medium High Max

Occupancy Is Not Limiting Kernel Performance

The kernel's block size, register usage, and shared memory usage allow it to fully utilize all warps on the GPU. More...

Variable	Achieved	Theoretical	Device Limit	Grid Size: [10000,6,9] (540000 blocks)Block Size: [16,16,1]
Occupancy Per SM				
Active Blocks		8	32	0 4 8 12 16 20 24 28 32
Active Warps	62.09	64	64	0 8 16 24 32 40 48 56 64
Active Threads		2048	2048	0 256 512 768 1024 1280 1536 1792 2048
Occupancy	97%	100%	100%	0% 25% 50% 75% 100%
Warps				
Threads/Block		256	1024	0 128 256 384 512 640 768 896 1024
Warps/Block		8	32	0 4 8 12 16 20 24 28 32
Block Limit		8	32	0 4 8 12 16 20 24 28 32
Registers				
Registers/Thread		30	65536	0 8192 16384 24576 32768 40960 49152 57344 65536
Registers/Block		8192	65536	0 16k 32k 48k 64k
Block Limit		8	32	0 4 8 12 16 20 24 28 32
Shared Memory				
Shared Memory/Block		1700	98304	0 32k 64k 96k
Block Limit		54	32	0 4 8 12 16 20 24 28 32



NVVP Analysis: The kernel seems to be limited by the amount of math done per call but that is because we are mapping each thread to one output. The math behind that is quite large. Maybe if we mapped every thread to an input, the runtime would be faster.

Optimization 6: Multiple Kernel implementations for different layer sizes

As we were doing our optimizations, we learned that different layer sizes have faster times with certain kernels. So far we had 3 different promising kernels so we decided to tune the tile widths and combinations of the 3 kernels that we saw some good op times in. During this process, inputs of each layer size was paired with a different kernel to sweep every combination of 2 kernels there is. On top of this, we adjusted the tile widths for the optimal performance of the combinations.

We know that the fused matrix multiply w/constant memory did the best with the second pass because it can handle the larger amount of layers with smaller inputs to take advantage of the ability to coalesce memory. On the other hand, the shared memory convolution worked best with the first pass as the input sizes were larger and there were less channels. We wanted to know if there was an optimal combination of kernels for each pass. To test this, we ran the following combinations:

Kernel type->first pass (6) |-second pass (16) V	Fused Matrix Multiply w/constant memory	Restrict and unroll Convolution	Shared memory convolution
Fused Matrix Multiply w/constant memory:	.0204, .0156 (TILEWIDTHS 16)	.0057, .0159 (TILE WIDTHS 16,24)	.0051, .0159 (TILE WIDTHS(24,16)

Restrict and unroll convolution	.01, .023 (TILEWIDTHS 16, 8)	.0066, .021(TILEWIDTHS 16,8)	.0058, .017 (TILEWIDTHS 16,24)
Shared memory convolution	.01, .18 (TILEWIDTHS 24,16)	.0056, .016 (TILEWIDTHS 16,24)	0.006718, 0.034986 (TILEWIDTH 16)

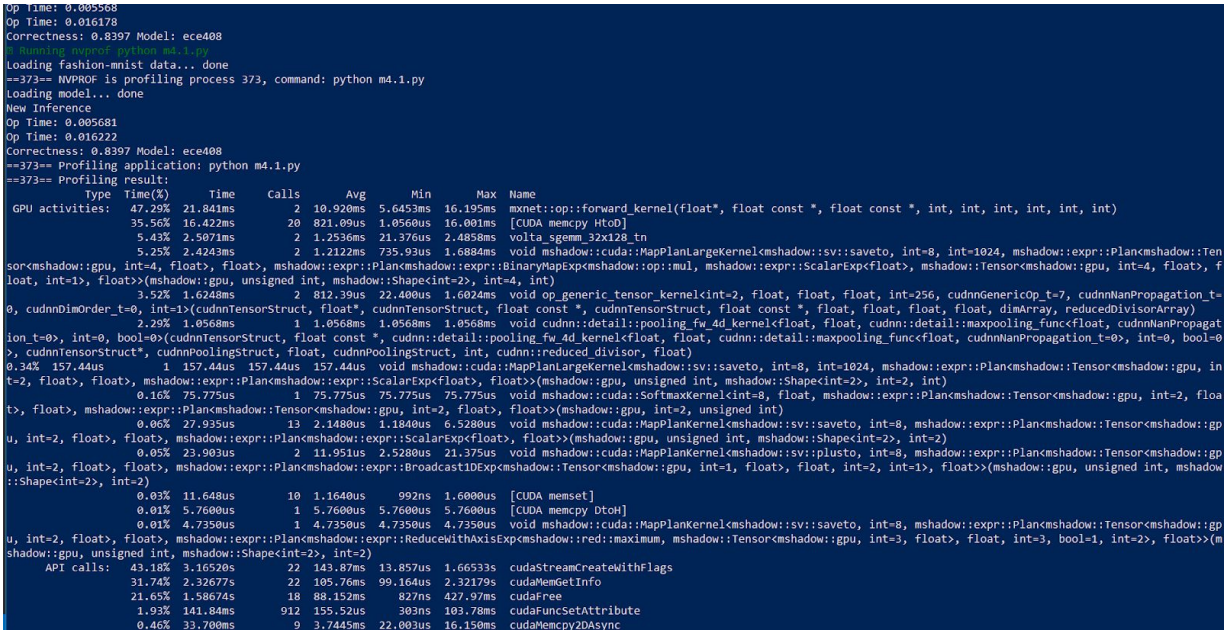
Restrict and unroll convolution (1st) Fused Matrix Multiply w/constant mem (2nd)

```
$ Running nvprof python m4.1.py
Loading fashion-mnist data... done
==372== NVPROF is profiling process 372, command: python m4.1.py
Loading model... done
New Inference
Op Time: 0.005185
Op Time: 0.015950
Correctness: 0.8397 Model: ece408
==372== Profiling application: python m4.1.py
==372== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   36.15%  16.370ms        20  818.52us  1.0560us  15.941ms  [CUDA memcpy HtoD]
                   35.11%  15.897ms         1  15.897ms  15.897ms  15.897ms  mxnet::op::forward_kernel2(float*, float*, int, int, int, int, int, int)
                   11.25%  5.0953ms         1  5.0953ms  5.0953ms  5.0953ms  mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)
                    5.54%  2.5092ms         2  1.2546ms  21.472us  2.4877ms  volta_sgemm_32x128_tn
                    5.33%  2.4137ms         2  1.2069ms  733.69us  1.6800ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tenso
r<mshadow::gpu, int=4, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4, float>, float
, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                    3.59%  1.6248ms         2  812.42us  21.727us  1.6031ms  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, float, dimArray, reducedDivisorArray)
                    2.33%  1.0546ms         1  1.0546ms  1.0546ms  1.0546ms  void cudnn::detail::pooling_fw_4d_kernel<float, float, cudnn::detail::maxpooling_func<float, cudnnNanPropagatio
n_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>, c
udnnTensorStruct*, cudnnPoolingStruct, float, cudnnPoolingStruct, int, cudnn::reduced_divisor, float)
                    0.35%  158.50us         1  158.50us  158.50us  158.50us  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tenso
r<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)
                    0.17%  75.200us         1  75.200us  75.200us  75.200us  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)
                    0.06%  27.839us        13  2.1410us  1.1830us  6.4640us  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)
                    0.05%  24.064us         2  12.032us  2.5280us  21.536us  void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu,
 int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::Broadcast1DExp<mshadow::Tensor<mshadow::gpu, int=1, float>, float, int=2, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Sh
ape<int=2>, int=2)
                    0.02%  11.199us        10  1.1190us     991ns  1.7280us  [CUDA memset]
                    0.01%  5.9840us         2  2.9920us  2.8800us  3.1040us  [CUDA memcpy DtoD]
                    0.01%  5.8880us         1  5.8880us  5.8880us  5.8880us  [CUDA memcpy DtoH]
                    0.01%  4.8000us         1  4.8000us  4.8000us  4.8000us  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::ReduceWithAxisExp<mshadow::red::maximum, mshadow::Tensor<mshadow::gpu, int=3, float>, float, int=3, bool=1, int=2>, float>>(mshad
ow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
      API calls:   42.30%  3.04589s        22  138.45ms
```

Shared mem convolution (1st), Fused Matrix Multiply w/constant mem (2nd)

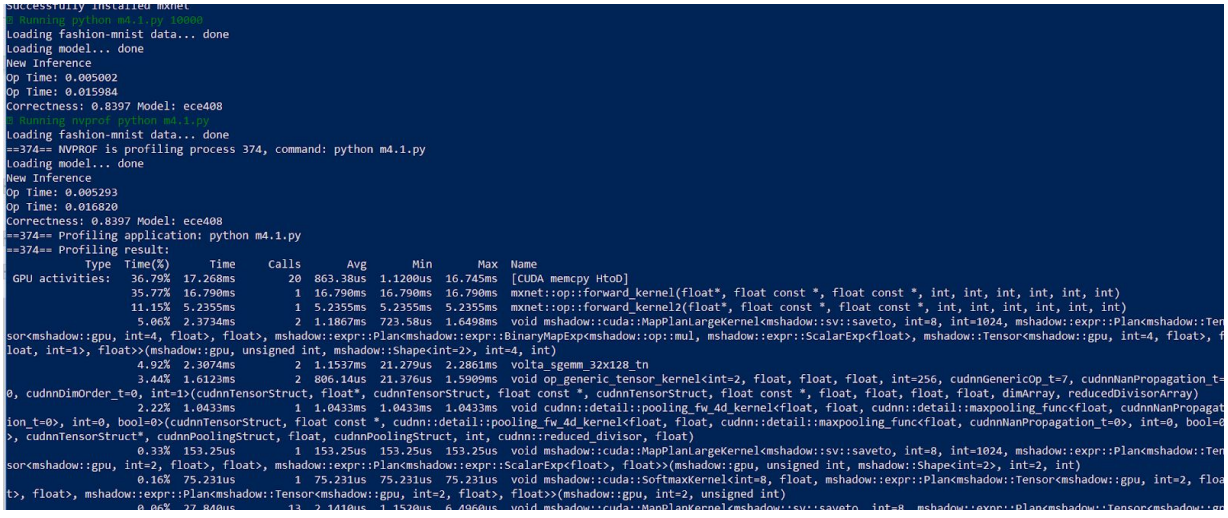
```
$ Running nvprof python m4.1.py
Loading fashion-mnist data... done
==373== NVPROF is profiling process 373, command: python m4.1.py
Loading model... done
New Inference
Op Time: 0.005703
Op Time: 0.015948
Correctness: 0.8397 Model: ece408
==373== Profiling application: python m4.1.py
==373== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   36.05%  16.600ms        20  829.99us  1.0880us  16.051ms  [CUDA memcpy HtoD]
                   34.52%  15.896ms         1  15.896ms  15.896ms  15.896ms  mxnet::op::forward_kernel2(float*, float*, int, int, int, int, int, int)
                   12.24%  5.6374ms         1  5.6374ms  5.6374ms  5.6374ms  mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)
                    5.44%  2.5066ms         2  1.2533ms  21.504us  2.4851ms  volta_sgemm_32x128_tn
                    5.25%  2.4151ms         2  1.2076ms  730.81us  1.6843ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tenso
r<mshadow::gpu, int=4, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4, float>, float
, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                    3.52%  1.6217ms         2  810.83us  22.112us  1.5995ms  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, float, dimArray, reducedDivisorArray)
                    2.28%  1.0514ms         1  1.0514ms  1.0514ms  1.0514ms  void cudnn::detail::pooling_fw_4d_kernel<float, float, cudnn::detail::maxpooling_func<float, cudnnNanPropagatio
n_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>, c
udnnTensorStruct*, cudnnPoolingStruct, float, cudnnPoolingStruct, int, cudnn::reduced_divisor, float)
                    0.34%  158.72us         1  158.72us  158.72us  158.72us  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tenso
r<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)
                    0.16%  75.167us         1  75.167us  75.167us  75.167us  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)
                    0.06%  27.936us        13  2.1480us  1.1840us  6.5280us  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)
                    0.05%  24.096us         2  12.048us  2.4960us  21.600us  void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu,
 int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::Broadcast1DExp<mshadow::Tensor<mshadow::gpu, int=1, float>, float, int=2, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Sh
ape<int=2>, int=2)
                    0.03%  12.255us        10  1.2250us     992ns  1.9840us  [CUDA memset]
                    0.02%  7.8720us         1  7.8720us  7.8720us  7.8720us  [CUDA memcpy DtoH]
                    0.01%  5.0560us         2  2.5280us  2.4640us  2.5920us  [CUDA memcpy DtoD]
                    0.01%  4.8000us         1  4.8000us  4.8000us  4.8000us  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::ReduceWithAxisExp<mshadow::red::maximum, mshadow::Tensor<mshadow::gpu, int=3, float>, float, int=3, bool=1, int=2>, float>>(mshad
ow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
      API calls:   43.56%  3.08525s        22  140.24ms
14.131us  1.58313s  cudaStreamCreateWithFlags
                   31.89%  2.25827s        22  102.65ms  95.054us  2.25328s  cudaMemGetInfo
                   22.19%  1.57184s        18  87.324ms     880ns  427.44ms  cudaFree
                    0.57%  40.468ms       216  187.35us     894ns  38.828ms  cudaEventCreateWithFlags
```

Shared mem convolution (1st) with Restrict tuning (2nd)



```
Op Time: 0.005568
Op Time: 0.016178
Correctness: 0.8397 Model: ece408
# Running nvprof python m4.1.py
Loading fashion-mnist data... done
==373== NVPROF is profiling process 373, command: python m4.1.py
Loading model... done
New Inference
Op Time: 0.005681
Op Time: 0.016222
Correctness: 0.8397 Model: ece408
==373== Profiling application: python m4.1.py
==373== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   47.29%  21.841ms         2  10.920ms  5.6453ms  16.195ms  mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)
                   35.56%  16.422ms        20  821.09us  1.0560us  16.001ms  [CUDA memcpy HtoD]
                    5.43%  2.5071ms         2  1.2536ms  21.376us  2.4858ms  volta_sgemm_32x128_tn
                    5.25%  2.4243ms         2  1.2122ms  735.93us  1.6884ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Ten
sor<mshadow::gpu, int=4, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4, float>, f
loat, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                    3.52%  1.6248ms         2  812.39us  22.400us  1.6024ms  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)
                    2.29%  1.0568ms         1  1.0568ms  1.0568ms  1.0568ms  void cudnn::detail::pooling_fw_4d_kernel<float, float, cudnn::detail::maxpooling_func<float, cudnnNanPropagat
ion_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)
0.34%  157.44us         1  157.44us  157.44us  157.44us  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, in
t=2, float>, float>, mshadow::expr::Plan<mshadow::expr::ScalarExp<float>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2, int)
                    0.16%  75.775us         1  75.775us  75.775us  75.775us  void mshadow::cuda::SoftmaxKernel<int=8, float, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, floa
t>, float>, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>>(mshadow::gpu, int=2, unsigned int)
                    0.06%  27.935us        13  2.1480us  1.1840us  6.5280us  void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gp
u, int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::ScalarExp<float>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
                    0.05%  23.903us         2  11.951us  2.5280us  21.375us  void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gp
u, int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::Broadcast1DExp<mshadow::Tensor<mshadow::gpu, int=1, float>, float, int=2, int=1>, float>>(mshadow::gpu, unsigned int, mshadow
::Shape<int=2>, int=2)
                    0.03%  11.648us        10  1.1640us     992ns  1.6000us  [CUDA memset]
                    0.01%  5.7600us         1  5.7600us  5.7600us  5.7600us  [CUDA memcpy DtoH]
                    0.01%  4.7350us         1  4.7350us  4.7350us  4.7350us  void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gp
u, int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::ReduceWithAxisExp<mshadow::red::maximum, mshadow::Tensor<mshadow::gpu, int=3, float>, float, int=3, bool=1, int=2>, float>>(m
shadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
      API calls:   43.18%  3.16520s        22  143.87ms  13.857us  1.66533s  cudaStreamCreateWithFlags
                   31.74%  2.32677s        22  105.76ms  99.164us  2.32179s  cudaMemGetInfo
                   21.65%  1.58674s        18  88.152ms     827ns  427.97ms  cudaFree
                    1.93%  141.84ms       912  155.52us     303ns  103.78ms  cudaFuncSetAttribute
                    0.46%  33.700ms         9  3.7445ms  22.003us  16.150ms  cudaMemcpy2DAsync
```

Restrict Unroll tuning Convolution(1st) Shared mem conv (2nd)



```
Successfully installed mxnet
# Running python m4.1.py 10000
Loading fashion-mnist data... done
Loading model... done
New Inference
Op Time: 0.005002
Op Time: 0.015984
Correctness: 0.8397 Model: ece408
# Running nvprof python m4.1.py
Loading fashion-mnist data... done
==374== NVPROF is profiling process 374, command: python m4.1.py
Loading model... done
New Inference
Op Time: 0.005293
Op Time: 0.016820
Correctness: 0.8397 Model: ece408
==374== Profiling application: python m4.1.py
==374== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   36.79%  17.268ms        20  863.38us  1.1200us  16.745ms  [CUDA memcpy HtoD]
                   35.77%  16.790ms         1  16.790ms  16.790ms  16.790ms  mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)
                   11.15%  5.2355ms         1  5.2355ms  5.2355ms  5.2355ms  mxnet::op::forward_kernel2(float*, float const *, float const *, int, int, int, int, int, int)
                    5.06%  2.3734ms         2  1.1867ms  723.58us  1.6498ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Ten
sor<mshadow::gpu, int=4, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4, float>, f
loat, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                    4.92%  2.3074ms         2  1.1537ms  21.279us  2.2861ms  volta_sgemm_32x128_tn
                    3.44%  1.6123ms         2  806.14us  21.376us  1.5909ms  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)
                    2.22%  1.0433ms         1  1.0433ms  1.0433ms  1.0433ms  void cudnn::detail::pooling_fw_4d_kernel<float, float, cudnn::detail::maxpooling_func<float, cudnnNanPropagat
ion_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)
                    0.33%  153.25us         1  153.25us  153.25us  153.25us  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Ten
sor<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)
                    0.16%  75.231us         1  75.231us  75.231us  75.231us  void mshadow::cuda::SoftmaxKernel<int=8, float, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, floa
t>, float>, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>>(mshadow::gpu, int=2, unsigned int)
                    0.06%  27.840us        13  2.1410us  1.1520us  6.4960us  void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gp
```

Fused Matrix Multiply w/constant mem (1st) Restrict and unroll convolution (2nd)

```
Loading model... done
New Inference
Op Time: 0.010055
Op Time: 0.017861
Correctness: 0.8397 Model: ece408
$ Running nvprof python m4.1.py
Loading fashion-mnist data... done
==373== NVPROF is profiling process 373, command: python m4.1.py
Loading model... done
New Inference
Op Time: 0.010296
Op Time: 0.017905
Correctness: 0.8397 Model: ece408
==373== Profiling application: python m4.1.py
==373== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   34.09%  17.861ms         1  17.861ms  17.861ms  17.861ms  mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)
                   31.29%  16.392ms        20  819.61us  1.1200us  16.006ms  [CUDA memcpy HtoD]
                   19.51%  10.223ms         1  10.223ms  10.223ms  10.223ms  mxnet::op::forward_kernel2(float*, float*, int, int, int, int, int, int)
                    4.79%  2.5079ms         2  1.2540ms  21.408us  2.4865ms  volta_sgemm_32x128_tn
                    4.62%  2.4220ms         2  1.2110ms  741.27us  1.6808ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Ten
sor<mshadow::gpu, int=4, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4, float>, f
loat, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                    3.10%  1.6265ms         2  813.23us  22.592us  1.6039ms  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)
                    2.00%  1.0476ms         1  1.0476ms  1.0476ms  1.0476ms  void cudnn::detail::pooling_fw_4d_kernel<float, float, cudnn::detail::maxpooling_func<float, cudnnNanPropagat
ion_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)
                    0.31%  160.77us         1  160.77us  160.77us  160.77us  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Ten
sor<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)
                    0.14%  75.231us         1  75.231us  75.231us  75.231us  void mshadow::cuda::SoftmaxKernel<int=8, float, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, floa
t>, float>, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>>(mshadow::gpu, int=2, unsigned int)
                    0.05%  27.903us        13  2.1460us  1.1840us  6.4960us  void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gp
u, int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::ScalarExp<float>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
                    0.05%  23.872us         2  11.936us  2.5600us  21.312us  void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gp
u, int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::Broadcast1DExp<mshadow::Tensor<mshadow::gpu, int=1, float>, float, int=2, int=1>, float>>(mshadow::gpu, unsigned int, mshadow
::Shape<int=2>, int=2)
                    0.02%  11.744us        10  1.1740us     992ns  2.1440us  [CUDA memset]
                    0.01%  5.8880us         1  5.8880us  5.8880us  5.8880us  [CUDA memcpy DtoH]
                    0.01%  5.2800us         2  2.6400us  2.3680us  2.9120us  [CUDA memcpy DtoD]
                    0.01%  4.9600us         1  4.9600us  4.9600us  4.9600us  void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gp
u, int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::ReduceWithAxisExp<mshadow::red::maximum, mshadow::Tensor<mshadow::gpu, int=3, float>, float, int=3, bool=1, int=2>, float>>(m
shadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
      API calls:   42.09%  3.02711s        22  137.60ms
```

Fused Matrix Multiply(1st) Convolution with restrict and unrolling tuning(2nd)

```
$ Running python m4.1.py 10000
Loading fashion-mnist data... done
Loading model... done
New Inference
Op Time: 0.010054
Op Time: 0.023503
Correctness: 0.8397 Model: ece408
$ Running nvprof python m4.1.py
Loading fashion-mnist data... done
==373== NVPROF is profiling process 373, command: python m4.1.py
Loading model... done
New Inference
Op Time: 0.010312
Op Time: 0.023672
Correctness: 0.8397 Model: ece408
==373== Profiling application: python m4.1.py
==373== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   40.00%  23.629ms         1  23.629ms  23.629ms  23.629ms  mxnet::op::forward_kernel(float*, float const *, float const *, int, int, int, int, int, int)
                   29.29%  17.298ms        20  864.92us  1.0880us  16.913ms  [CUDA memcpy HtoD]
                   17.31%  10.225ms         1  10.225ms  10.225ms  10.225ms  mxnet::op::forward_kernel2(float*, float*, int, int, int, int, int, int)
                    4.23%  2.4964ms         2  1.2482ms  21.568us  2.4749ms  volta_sgemm_32x128_tn
                    4.11%  2.4267ms         2  1.2133ms  734.04us  1.6927ms  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Ten
sor<mshadow::gpu, int=4, float>, float>, mshadow::expr::Plan<mshadow::expr::BinaryMapExp<mshadow::op::mul, mshadow::expr::ScalarExp<float>, mshadow::Tensor<mshadow::gpu, int=4, float>, f
loat, int=1>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=4, int)
                    2.75%  1.6239ms         2  811.93us  22.144us  1.6017ms  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)
                    1.79%  1.0561ms         1  1.0561ms  1.0561ms  1.0561ms  void cudnn::detail::pooling_fw_4d_kernel<float, float, cudnn::detail::maxpooling_func<float, cudnnNanPropagat
ion_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)
                    0.27%  158.72us         1  158.72us  158.72us  158.72us  void mshadow::cuda::MapPlanLargeKernel<mshadow::sv::saveto, int=8, int=1024, mshadow::expr::Plan<mshadow::Ten
sor<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)
                    0.13%  75.456us         1  75.456us  75.456us  75.456us  void mshadow::cuda::SoftmaxKernel<int=8, float, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, floa
t>, float>, mshadow::expr::Plan<mshadow::Tensor<mshadow::gpu, int=2, float>, float>>(mshadow::gpu, int=2, unsigned int)
                    0.05%  28.192us        13  2.1680us  1.1840us  6.5920us  void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gp
u, int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::ScalarExp<float>, float>>(mshadow::gpu, unsigned int, mshadow::Shape<int=2>, int=2)
                    0.04%  24.064us         2  12.032us  2.5600us  21.504us  void mshadow::cuda::MapPlanKernel<mshadow::sv::plusto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gp
u, int=2, float>, float>, mshadow::expr::Plan<mshadow::expr::Broadcast1DExp<mshadow::Tensor<mshadow::gpu, int=1, float>, float, int=2, int=1>, float>>(mshadow::gpu, unsigned int, mshadow
::Shape<int=2>, int=2)
                    0.02%  11.328us        10  1.1320us     992ns  1.6000us  [CUDA memset]
                    0.01%  5.9200us         1  5.9200us  5.9200us  5.9200us  [CUDA memcpy DtoH]
                    0.01%  5.2800us         2  2.6400us  2.4960us  2.7840us  [CUDA memcpy DtoD]
                    0.01%  4.6400us         1  4.6400us  4.6400us  4.6400us  void mshadow::cuda::MapPlanKernel<mshadow::sv::saveto, int=8, mshadow::expr::Plan<mshadow::Tensor<mshadow::gp
```

NVVP

Kernel fusion pass

Results

	Transactions	Bandwidth	Utilization
Shared Memory			
Shared Loads	331854593	8,894.192 GB/s	
Shared Stores	15175163	406.717 GB/s	
Shared Total	347029756	9,300.908 GB/s	Idle Low Medium High Max
L2 Cache			
Reads	26882050	180.12 GB/s	
Writes	18480247	123.824 GB/s	
Total	45362297	303.944 GB/s	Idle Low Medium High Max
Unified Cache			
Local Loads	0	0 B/s	
Local Stores	0	0 B/s	
Global Loads	26909714	180.305 GB/s	
Global Stores	18480000	123.823 GB/s	
Texture Reads	235145991	6,302.259 GB/s	
Unified Total	280535705	6,606.387 GB/s	Idle Low Medium High Max
Device Memory			
Reads	33594036	225.092 GB/s	
Writes	18516349	124.066 GB/s	
Total	52110385	349.159 GB/s	Idle Low Medium High Max
System Memory [PCIe configuration: Gen3 x8, 8 Gbit/s]			
Reads	0	0 B/s	Idle Low Medium High Max
Writes	5	33.501 kB/s	Idle Low Medium High Max

Occupancy Is Not Limiting Kernel Performance

The kernel's block size, register usage, and shared memory usage allow it to fully utilize all warps on the GPU. More...

Variable	Achieved	Theoretical	Device Limit	Grid Size: [10000,6,9] (540000 blocks)Block Size: [16,16,1]
Occupancy Per SM				
Active Blocks		8	32	0 4 8 12 16 20 24 28 32
Active Warps	62.09	64	64	0 8 16 24 32 40 48 56 64
Active Threads		2048	2048	0 256 512 768 1024 1280 1536 1792 2048
Occupancy	97%	100%	100%	0% 25% 50% 75% 100%
Warps				
Threads/Block		256	1024	0 128 256 384 512 640 768 896 1024
Warps/Block		8	32	0 4 8 12 16 20 24 28 32
Block Limit		8	32	0 4 8 12 16 20 24 28 32
Registers				
Registers/Thread		30	65536	0 8192 16384 24576 32768 40960 49152 57344 65536
Registers/Block		8192	65536	0 16k 32k 48k 64k
Block Limit		8	32	0 4 8 12 16 20 24 28 32
Shared Memory				
Shared Memory/Block		1700	98304	0 32k 64k 96k
Block Limit		54	32	0 4 8 12 16 20 24 28 32

Shared Mem convolution pass

Memory Bandwidth And Utilization

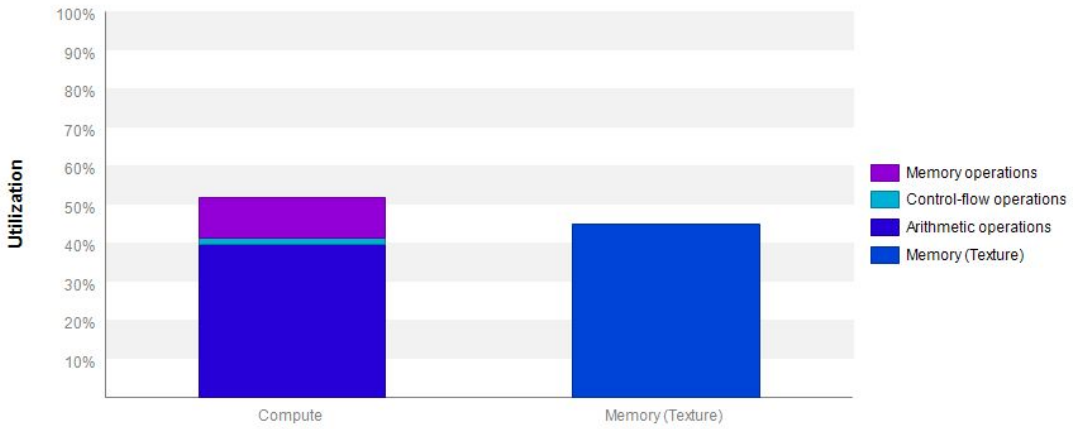
The following table shows the memory bandwidth used by this kernel for the various types of memory on the device. The table also shows the utilization of each memory type relative to the maximum throughput supported by the memory. More...

	Transactions	Bandwidth	Utilization
Shared Memory			
Shared Loads	330191009	6,995.915 GB/s	
Shared Stores	15161650	321.237 GB/s	
Shared Total	345352659	7,317.152 GB/s	Idle Low Medium High Max
L2 Cache			
Reads	26882967	142.396 GB/s	
Writes	18480016	97.886 GB/s	
Total	45362983	240.282 GB/s	Idle Low Medium High Max
Unified Cache			
Local Loads	0	0 B/s	
Local Stores	0	0 B/s	
Global Loads	26891524	142.441 GB/s	
Global Stores	18480000	97.886 GB/s	
Texture Reads	235145723	4,982.145 GB/s	
Unified Total	280517247	5,222.472 GB/s	Idle Low Medium High Max
Device Memory			
Reads	33594060	177.943 GB/s	
Writes	18517670	98.086 GB/s	
Total	52111730	276.029 GB/s	Idle Low Medium High Max
System Memory [PCIe configuration: Gen3 x8, 8 Gbit/s]			
Reads	0	0 B/s	Idle Low Medium High Max
Writes	5	26.484 kB/s	Idle Low Medium High Max

Occupancy Per SM				
Active Blocks		6	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Active Warps	46.65	48	64	0 5 10 15 20 25 30 35 40 45 50 55 60 64
Active Threads		1536	2048	0 256 512 768 1024 1280 1536 1792 2048
Occupancy	72.9%	75%	100%	0% 15% 30% 45% 60% 75% 90% 100%
Warps				
Threads/Block		256	1024	0 128 256 384 512 640 768 896 1024
Warps/Block		8	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Block Limit		8	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Registers				
Registers/Thread		39	65536	0 8192 16384 24576 32768 40960 49152 57344 65536
Registers/Block		10240	65536	0 16k 32k 48k 64k
Block Limit		6	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32
Shared Memory				
Shared Memory/Block		1700	98304	0 16k 32k 48k 64k 80k 96k
Block Limit		54	32	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32

Kernel Performance Is Bound By Instruction And Memory Latency

This kernel exhibits low compute throughput and memory bandwidth utilization relative to the peak performance of "TITAN V". These utilization levels indicate that the performance of the kernel is most likely limited by the latency of arithmetic or memory operations. Achieved compute throughput and/or memory bandwidth below 60% of peak typically indicates latency issues.



Analysis: This combination actually uses less shared memory than the fusion kernel alone but runs faster perhaps because of the smaller amount of arithmetic operations. There seems to be a memory latency issue which might be a result of the redundant loads from X and the many loads we have in general even if they are coalesced/avoid bank conflicts in most cases. The NVVPs are very similar to that of the single kernels. Both the block limits have been reached and the use of shared memory is pretty high for both of the kernels. When you compare this to the NVVPs for the respective kernel calls of the original however, we see that the pass and kernel combinations had better overall usage of resources and the fastest runtime.

Additional insights:

Something we've learned about CUDA is that you cannot dynamically allocate shared memory unless you declare it as extern. When we were fusing the matrix multiply kernel, we tried to change the tile width based on an if statement before the kernel call. After further research, we found that when the program runs, cuda only supports a single dynamically declared memory allocation for each block. If we need more, we would have needed to use pointers within that allocation

When we originally tried Unrolled matrix multiply: Convolution operation essentially performs dot products between the filters and local regions of the input**.** By unrolling the forward pass into a large matrix, we can run through all the calculations in bulk, leaving most of the work in the inner for loops of the channels. Additionally, the memory is all stored together giving potential for memory coalescing within that kernel. The downside to matrix multiply is that there are a lot of redundant loads from X. This implies that if there are many channels and layers, these could outweigh the fact that all the memory for this kernel is bunched together.

We expected the unroll to really help with the memory usage because we had the opportunity to coalesce. However, we realized the the transfers of memory were actually incredibly heavy from this. On top of this, the redundant loads from X also hit our runtime hard.

Work division (All optimizations): We made sure everyone did 2 optimizations each and split up the work evenly across the first 3 milestones. Only Kevin was able to install cuda toolkit so he got everyone the nvvp results. Overall, the work was split up very evenly.

Works cited (All optimizations):

H, Albert. *Devtalk.nvidia.com*. N.p., 23 Dec. 2016. Web. 25 Apr. 2019.

Lee, Daren, et al. “CUDA Optimization Strategies for Compute- and Memory-Bound Neuroimaging Algorithms.” *Computer Methods and Programs in Biomedicine*, U.S. National Library of Medicine, June 2012, www.ncbi.nlm.nih.gov/pmc/articles/PMC3262956/.Web. 30 April 2019.

Shafkat, Irhum. “Intuitively Understanding Convolutions for Deep Learning.” *Towards Data Science*, Towards Data Science, 1 June 2018, towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1.

Woolley, Cliff. “GPU Optimization Fundamentals.” *Bluewaters Ncsa*, 2013, bluewaters.ncsa.illinois.edu.Web. 21 April 2019.

"An Effective Method for Better Power Efficiency on Multithreaded GPU." *Kernel Fusion*. IEEE Computer Society, 2010. Web. 17 April 2019.

"Loop Unrolling." *GeeksforGeeks*. N.p., 19 Feb. 2018. Web. 3 May 2019.

http://homepages.math.uic.edu/~jan/mcs572/memory_coalescing.pdf

http://cseweb.ucsd.edu/classes/wi12/cse260-a/Lectures/Lec08.pdf